# Accelerating Batch Active Learning Using Continual Learning Techniques

**Anonymous authors**
**Paper under double-blind review**

## Abstract

A major problem with Active Learning (AL) is high training costs since models are typically retrained from scratch after every query round. We start by demonstrating that standard AL on neural networks with warm starting fails, both to accelerate training and to avoid catastrophic forgetting when using fine-tuning over AL query rounds. We then develop a new class of techniques, circumventing this problem, by biasing further training towards previously labeled sets. We accomplish this by employing existing, and developing novel, replay-based Continual Learning (CL) algorithms that are effective at quickly learning the new without forgetting the old, especially when data comes from an evolving distribution. We call this paradigm *"Continual Active Learning" (CAL)*. We show CAL achieves significant speedups using a plethora of replay schemes that use model distillation and that select diverse/uncertain points from the history. We conduct experiments across many data domains, including natural language, vision, medical imaging, and computational biology, each with different neural architectures and dataset sizes. CAL consistently provides a ∼3x reduction in training time, while retaining performance and out-of-distribution robustness, showing its wide applicability.

## 1 Introduction

While neural networks have been successful in a variety of different supervised settings, most such approaches are labeled-data hungry and require significant computation. From a large pool of unlabeled data, active learning (AL) selects subsets of points to label by imparting the learner with the ability to query a human annotator. Such methods incrementally add points to the labeled pool by repeatedly: (1) training a model from scratch on the current labeled pool and (2) using some measure of model uncertainty and/or diversity to select a set of points to query the annotator (Settles, 2009; 2011; Wei et al., 2015; Ash et al., 2020; Killamsetty et al., 2021a). AL has been shown to reduce the amount of training data required but can be computationally expensive since it requires retraining a model, typically from scratch, after each query round.

A *simple* solution is to warm start the model parameters between query rounds. However, the observed speedups tend to still be limited since the model must make several passes through an ever-increasing pool of data. Moreover, warm starting alone in some cases can hurt generalization, as discussed in Ash & Adams (2020) and Beck et al. (2021). Another extension to this is to solely train on the newly labeled batch of examples to avoid re-initialization. However, as we show in Section 4, naive fine-tuning fails to retain accuracy on previously seen examples since the distribution of the query pool may drastically change with each round.

This problem of *catastrophic forgetting* while incrementally learning from a series of new tasks with shifting distributions is a central question in another paradigm called Continual Learning (CL) (French, 1999; McCloskey & Cohen, 1989; McClelland et al., 1995; Kirkpatrick et al., 2017c). CL has recently gained popularity, and many algorithms have been introduced to allow models to quickly adapt to new tasks without forgetting (Riemer et al., 2018; Lopez-Paz & Ranzato, 2017; Chaudhry et al., 2019; Aljundi et al., 2019b; Chaudhry et al., 2020; Kirkpatrick et al., 2017b).

In this work, we propose Continual Active Learning (CAL)[1], which applies Continual Learning strategies to accelerate batch Active Learning. In CAL, we apply CL to enable the model to learn the newly labeled points

[1]There have been papers published with titles containing the "Continual Active Learning" phrase, but these do not merge **Continual** Learning with **Active Learning** as we do, hence our name.

without forgetting previously labeled points while using past samples efficiently using *replay-based* methods. As such, we observe that CAL attains significant training time speedups over standard AL. This is beneficial for the following reasons: **(1):** As neural networks swell (Shoeybi et al., 2019), so do the environmental and financial model training costs (Bender et al., 2021; Dhar, 2020; Schwartz et al., 2020). Reducing the number of gradient updates required for AL will help mitigate such costs, especially with large-scale models. **(2):** Reducing AL computational requirements makes AL more accessible for edge computing, IoT, and low-resource device deployment (Senzaki & Hamelain, 2021) such as with federated learning (Li et al., 2020). **(3):** Developing new AL algorithms/acquisition functions, or searching for architectures as done with NAS/AutoML that are well-suited *specifically* for AL, can require hundreds or even thousands of runs. Since CAL's speedups are agnostic to the AL algorithm and the neural architecture, such experiments can be significantly sped up. Overall, the importance of speeding machine learning training processes is well recognized, as evidenced by the plethora of efforts in the computing systems community (Jia et al., 2022; Zhang et al., 2017; Zheng et al., 2022).

In addition, CAL demonstrates another practical application for CL methods. Many settings used to benchmark CL methods in recent work are somewhat contrived (Farquhar & Gal, 2018; Wallingford et al., 2023). Most CL work considers the class/domain incremental setting, where only the samples that belong to a subset of the set of classes/domains of the original dataset are available to the model at any given time. This setting need not be the only benchmark upon which CL methods are evaluated. We suggest that the evaluation of future new CL algorithms should be determined not only on traditional CL evaluation schemes and benchmarks but also on their performance in the CAL setting.

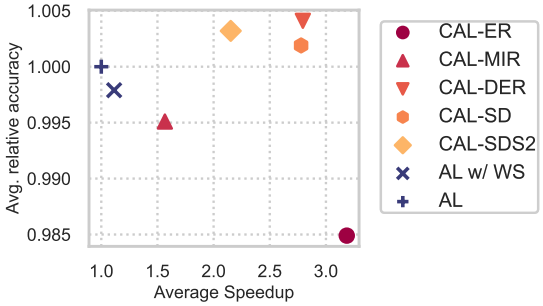


Figure 1: Summary of our results: Benchmarking average relative accuracy vs. speedup for the proposed methods (*solid markers*) against the baseline. Each point is averaged across datasets and labeling budgets; *top-right region is desiderata.* This shows that proposed CAL methods (particularly CAL-DER/SD/SDS2) have equivalent performance relative to the baseline (if not better) on multiple datasets and across different budgets, all while providing significant speedups.

In the present work, we are not using CL to improve or change AL querying strategies. We view this as both a strength of the present work and an opportunity for future work. Firstly, it is a strength of our present work since any AL query strategy, both old and new, can in principle be applied in the CAL setting, both for speeding up the AL strategy and offering, as argued above, a test bed for CL. Indeed, a major AL research challenge includes combining AL with other techniques, as we believe we have done herein. Secondly, it is an opportunity since there is nothing inherent in the definition of CL that precludes the CL tasks from being dependent on the model. There may be a way for CL to open doors to intrinsically new AL querying policies of attaining new batches of unlabeled data. This we leave to the future. Our present *core goal is to accelerate batch AL training via CL techniques while preserving accuracy.*

To the best of our knowledge, this application of CL algorithms to accelerate batch AL has never been explored. Our contributions can be summarized as follows: **(1)** We first demonstrate that batch active learning techniques can benefit from continual learning techniques, and their merger creates a new class of techniques that we propose to call the "CAL framework." **(2)** We benchmark several existing CL methods (CAL-ER, CAL-DER, CAL-MIR) as well as novel methods (CAL-SD, CAL-SDS2) and evaluate them on diverse datasets based on the accuracy/speedup they can attain over standard AL. **(3)** We study speedup/performance trade-offs on datasets that vary in modality (natural language, vision, medical imaging, and computational biology), neural architecture with varying degrees of computation (Transformers/CNNs/MLPs), data scale (including some larger datasets, one having 2M samples), and class-balance. And **(4)**, lastly, we demonstrate that models trained with CAL and standard AL models behave similarly, in that both classes of models attain similar uncertainty scores on held-out datasets and achieve similar robustness performance on out-of-distribution data. Figure 1 summarizes our results, detailed later in the paper and greatly detailed in the appendices.

## 2 Related Work

Active learning has demonstrated label efficiency (Cohn et al., 1994; Wei et al., 2015; Killamsetty et al., 2021a; Ash et al., 2020) over passive learning. In addition, there has been extensive work on theoretical aspects of AL (Hanneke, 2009; 2007; Balcan et al., 2010) where Hanneke (2012) shows sample complexity advantages over passive learning in noise-free classifier learning for VC classes. Recently there has been an interest in speeding up active learning since most deep learning is computationally demanding.

Kirsch et al. (2019); Pinsler et al. (2019); Sener & Savarese (2018) aim to reduce the number of query iterations by having large query batch sizes. However, they do not exploit the learned models from previous rounds for the subsequent ones and are therefore complementary to CAL. Work such as Coleman et al. (2020a); Ertekin et al. (2007); Mayer & Timofte (2020); Zhu & Bento (2017) speeds up the selection of the new query set by appropriately restricting the search space or by using generative methods. This work can be easily integrated into our framework because CAL works on the training side of active learning, not on the query selection. On the other hand, Lewis & Catlett (1994); Coleman et al. (2020b); Yoo & Kweon (2019) use a smaller proxy model to reduce computation overhead, however, they still follow the standard active learning protocol, and therefore can be accelerated when integrated with CAL.

There exists work that explores continual/transfer learning and active learning in the same context. Perkonigg et al. (2021) propose an approach that allows active learning to be applied to data streams of medical images by introducing a module that detects domain shifts. This is quite distinct from our work: our work uses CL algorithms to prevent catastrophic forgetting and to accelerate learning. Zhou et al. (2021) study when standard active learning is used to fine-tune a pre-trained model, and employs transfer learning — this does not consider continual learning and active learning together, however, and is therefore not related to our work. Finally, Ayub & Fendley (2022) studies where a robot observes unlabeled data sampled from a shifting distribution, but does not explore active learning acceleration.

For preventing catastrophic forgetting, we mostly focus on replay-based algorithms that are state-of-the-art methods in CL. However, as demonstrated in Section 4 on how active learning rounds can be viewed in a continual learning context, one can apply other methods such as EWC (Kirkpatrick et al., 2017a), Bayesian divergence priors Li & Bilmes (2007), structural regularization (Li et al., 2021) or functional regularization (Titsias et al., 2020) as well.

The effect of warm-started model training on generalization and convergence speed has been explored by Ash & Adams (2020) which empirically demonstrates that a model that has been pretrained on a source dataset converges faster but exhibits worse generalization on a target dataset when compared to a randomly initialized model. However, that work only considers the setting where the source and target datasets are unbiased estimates of the same distribution. This is distinct from our work since the distributions we consider are all dependent on the model at each AL round. Furthermore, our work employs CL methods in addition to warm-starting, also not considered in Ash & Adams (2020).

## 3 Background

### 3.1 Batch Active Learning

Define $[n] = \{1, ..., n\}$, and let $\mathcal{X}$ and $\mathcal{Y}$ denote the input and output domains respectively. AL typically starts with an unlabeled dataset $\mathcal{U} = \{x_i\}_{i \in [n]}$, where each $x_i \in \mathcal{X}$. The AL setting allows the model $f$, with parameters $\theta$, to query a user for labels for any $x \in \mathcal{U}$, but the total number of labels is limited to a budget $b$, where $b \leq n$. Throughout the work, we consider classification tasks so the output of $f(x; \theta)$ is a probability distribution over classes. The goal of AL is to ensure that $f$ can attain low error when trained only on the set of $b$ labeled points.

Algorithm 1 details the general AL procedure. Lines 3-6 construct the seed set $\mathcal{D}_1$ by randomly sampling a subset of points from $\mathcal{U}$ and labeling them. Lines 7-14 iteratively expand the labeled set for $T$ rounds by training the model from a random initialization on $\mathcal{D}_t$ until convergence and selecting $b_t$ points (where $\sum_{t \in [T]} b_t = b$) from $\mathcal{U}$ based on some selection criteria that is dependent on $\theta_t$. The selection criteria generally

select samples based on model uncertainty and/or diversity (Lewis & Gale, 1994; Dagan & Engelson, 1995; Settles; Killamsetty et al., 2021a; Wei et al., 2015; Ash et al., 2020; Sener & Savarese, 2017). In this work, we primarily consider uncertainty sampling Lewis & Gale (1994); Dagan & Engelson (1995); Settles, though we also test other selection criteria in Section C in the Appendix.

**Uncertainty Sampling** is a widely-used practical AL method that selects $\mathcal{U}_t = \{x_1, \ldots, x_{b_t}\} \subseteq \mathcal{U}$ by choosing the samples that maximize a notion of model uncertainty. We consider entropy (Dagan & Engelson, 1995) as the uncertainty metric, so if $h_\theta(x) \triangleq -\sum_{i\in[k]} f(x;\theta)_i \log f(x;\theta)_i$, then $\mathcal{U}_{t+1} \in \underset{\mathcal{A}\subset\mathcal{U}:|\mathcal{A}|=b_t}{\operatorname{argmax}} \sum_{x\in\mathcal{A}} h_{\theta_t}(x)$.

**Algorithm 1** Batch Active Learning

1: **procedure** ActiveLearning($f$, $\mathcal{U}$, $b_{1:T}$, $T$)
2: $t \leftarrow 1$, $\mathfrak{L} \leftarrow \emptyset$ ▷ Initialize
3: $\mathcal{U}_t \sim \mathcal{U}$ ▷ Draw $b_1$ samples from $\mathcal{U}$
4: $\mathcal{D}_t \leftarrow \{(x_i, y_i) | x_i \in \mathcal{U}_t\}$ ▷ Provide labels
5: $\mathcal{U} \leftarrow \mathcal{U} \setminus \mathcal{U}_t$ ▷ Remove from unlabeled set
6: $\mathfrak{L} \leftarrow \mathfrak{L} \cup \mathcal{D}_t$ ▷ Add to labeled set
7: **while** $t \leq T$ **do**
8: Randomly initialize $\theta_{init}$
9: $\theta_t \leftarrow \text{Train}(f, \theta_{init}, \mathfrak{L})$
10: $\mathcal{U}_{t+1} \leftarrow \text{Select}(f, \theta_t, \mathcal{U}, b_t)$ ▷ Select $b_t$ points based on $\theta_t$
11: $\mathcal{D}_{t+1} \leftarrow \{(x_i, y_i) | x_i \in \mathcal{U}_t\}$
12: $\mathcal{U} \leftarrow \mathcal{U} \setminus \mathcal{U}_{t+1}$; $\mathfrak{L} \leftarrow \mathfrak{L} \cup \mathcal{D}_{t+1}$; $t \leftarrow t + 1$
13: $\theta_T \leftarrow \text{Train}(f, \theta_{init}, \mathfrak{L})$
14: return $\mathfrak{L}$, $\theta_T$

### 3.2 Continual Learning

We define $\mathcal{D}_{1:n} = \bigcup_{i\in[n]} \mathcal{D}_i$. In CL, the dataset consists of $T$ tasks $\{\mathcal{D}_1, ..., \mathcal{D}_T\}$ that are presented to the model sequentially, where $\mathcal{D}_t = \{(x_i, y_i)\}_{i\in N_t}$, $N_t$ are the task-$t$ sample indices, and $n_t = |N_t|$. At time $t \in [T]$, the data/label pairs are sampled from the current task $(x, y) \sim \mathcal{D}_t$, and the model has only limited access to the history $\mathcal{D}_{1:t-1}$. The CL objective is to efficiently adapt the model to $\mathcal{D}_t$ while ensuring performance on the history does not appreciably degrade. We focus on replay-based CL techniques that attempt to approximately solve CL optimization by using samples from $\mathcal{D}_{1:t-1}$ to regularize the model while adapting to $\mathcal{D}_t$. Please refer to appendix B for more details on CL.

Algorithm 2 outlines general replay-based CL, where the objective is to adapt $f$ parameterized by $\theta_0$ to $\mathcal{D}$ while using samples from the history $\mathcal{H}$. $\mathcal{B}_{\text{current}}$ consists of $m$ points randomly sampled from $\mathcal{D}$, and $\mathcal{B}_{\text{replay}}$ consists of $m^{(h)}$ points chosen based on a criterion that selects from $\mathcal{H}$. In line 6, $\theta_\tau$ is computed based on an update rule that utilizes both $\mathcal{B}_{\text{replay}}$ and $\mathcal{B}_{\text{current}}$. Memory-based CL techniques consider the problem of selecting which samples should be retained from $\mathcal{H}$, quite important when $\mathcal{D}_{1:T}$ is too large to store in memory or when $T$ is unknown (Aljundi et al., 2019b). However, this constraint does not apply to the CAL setting, so in the subsequent sections, we consider $\mathcal{H} = \mathcal{D}_{1:t-1}$. It is also possible to use purely regularization based CL techniques, and avoid any form of replay as considered in Kirkpatrick et al. (2017b). However, such methods are typically only practical in settings where it is not possible to store any samples from history, as they tend to underperform other techniques when this constraint is relaxed Buzzega et al. (2020).

**Algorithm 2** Continual Learning

1: **procedure** ContinualTrain($f$, $\theta_0$, $\mathcal{D}$, $\mathcal{H}$, $m$, $m^{(h)}$)
2: $\tau \leftarrow 0$ ▷ $\tau$ is a local iteration index.
3: **while** not converged **do**
4: $\tau \leftarrow \tau + 1$
5: $\mathcal{B}_{\text{current}} \leftarrow \{(x_i, y_i)\}_{i=1}^m \sim \mathcal{D}$ ▷ Sample from new task.
6: $\mathcal{B}_{\text{replay}} \leftarrow \text{Select}(f, \theta_{\tau-1}, \mathcal{H}, m^{(h)})$ ▷ Sample from history.
7: $\theta_\tau \leftarrow \text{Update}(f, \theta_{\tau-1}, \mathcal{B}_{\text{current}}, \mathcal{B}_{\text{replay}})$
8: return $\theta_\tau$

## 4 Blending Continual and Active Learning

A clear AL inefficiency is that the model $f$ is retrained from scratch on the entire labeled pool after every query round. One potential solution idea is to simply continue training the model only on the newly AL-queried samples and, via the process of warm starting, hope that history will not fade. Unfortunately for this approach, Figure 2 (top) shows that when the model is warm-start trained *only* on the task $t$ (samples labeled at AL round $t$ using entropy sampling), historical sample performance deteriorates precipitously while performance on the validation set flatlines. That is, at AL round $t$ (x-axis), we continue to train the model until convergence on task $t$ and track accuracy (y-axis) on each previous task and also on the validation set. The performance of task 1, after the initial drop, tracks that of the validation set, since task 1 is a model agnostic initial unbiased random subset query of the training data. The performance of task $i$, for $i > 1$,

however, each of which is the result of model-conditioned AL query, shows perilous historical forgetting. In the end, the model performs considerably worse on all of the historical tasks (aside from task 1) than the validation set, even though it has been trained on those tasks and not the validation set. This experiment suggests that: (1) the distribution of each AL-queried task $t > 1$ is different than the data distribution; (2) fine-tuning to task $t$ can result in catastrophic forgetting; and (3) techniques to combat catastrophic forgetting are necessary to effectively incorporate new information between successive AL rounds.

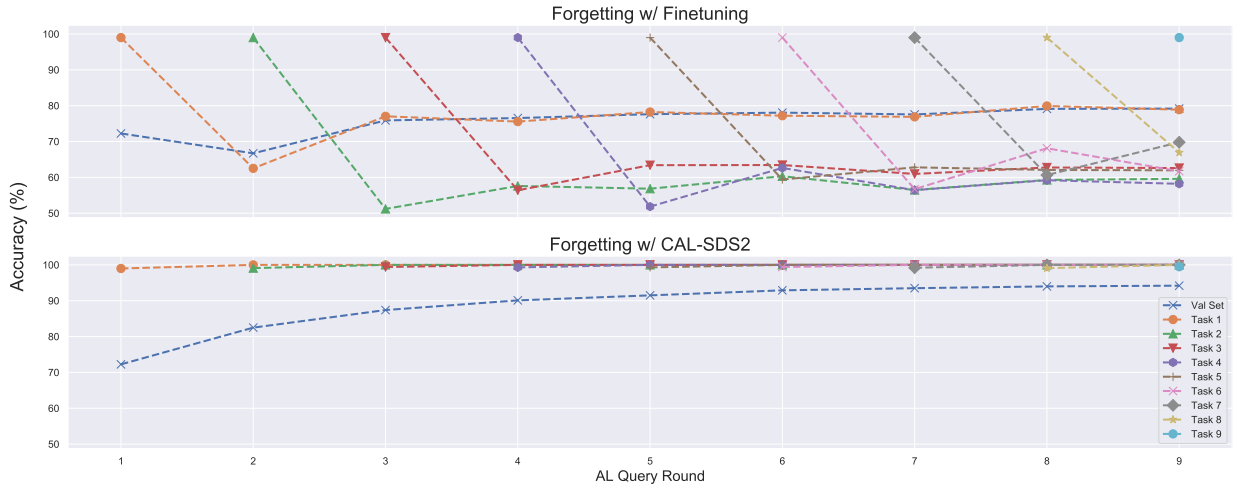


Figure 2: A ResNet-18's performance on CIFAR-10 on the validation set (blue X) and task $i$ for $i \in \{1, 2, \ldots, 9\}$ (other colors) after training the warm-started model and training just on the new task data (top) compared to efficiently adapting to newly labeled data with a CAL method (bottom). In the top plot, at each round a new 5% of the full dataset is added to the labeled pool and the previously trained model is further trained *just* on the new points (the isolated upper-right cyan point shows task 9's performance after round 9). After each round, the accuracy on the corresponding task does well, but all previous tasks' accuracies drop precipitously, demonstrating a form of catastrophic forgetting. In the bottom plot, the same setting is considered but the model is adapted to the new data with CAL-SDS2, one of the CAL methods. Also, the validation set performance has flatlined despite continued training on new data when finetuning, but continues to increase when CAL-SDS2 is employed. This demonstrates that naively fine-tuning a model to newly labeled points can work poorly.

The CAL approach, shown in Algorithm 3, uses CL techniques to ameliorate catastrophic forgetting. The key difference between CAL and Algorithm 1 is line 9. Instead of standard training, replay-based CL is used to adapt $f$ to $\mathcal{D}_t$ while retaining performance on $\mathcal{D}_{1:t-1}$. The speedup comes from two sources: (1) we are computing gradient updates only on a useful subset of the history $\mathcal{D}_{1:t-1}$ rather than all of it for reasonable choices of $m^{(h)}$; and (2) the model converges faster since it starts warm. The ability of CAL to combat catastrophic forgetting is shown in Figure Figure 2 (bottom). In the rest of the section, define $\mathcal{L}_c(\theta) \triangleq \mathbb{E}_{(x,y)\sim\mathcal{B}_{\text{current}}}\left[\ell(y, f(x; \theta))\right]$. We next define and compare several CAL methods and assess their performance based on their performance on the test set and the speedup they attain compared to standard AL.

**Algorithm 3** The general CAL approach.

1: **procedure** CAL($f$, $\mathcal{U}$, $b_{1:T}$, $T$, $m$, $m^{(h)}$)
2: $t \leftarrow 1$, $\mathfrak{L} \leftarrow \emptyset$ ▷ Initialize
3: $\mathcal{U}_t \sim \mathcal{U}$ ▷ Draw $b_1$ samples from $\mathcal{U}$
4: $\mathcal{D}_t \leftarrow \{(x_i, y_i) | x_i \in \mathcal{U}_t\}$ ▷ Provide labels
5: $\mathcal{U} \leftarrow \mathcal{U} \setminus \mathcal{U}_t$
6: $\mathfrak{L} \leftarrow \mathfrak{L} \cup \mathcal{D}_t$
7: Randomly initialize $\theta_0$
8: **while** $t \leq T$ **do**
9: $\theta_t \leftarrow$ ContinualTrain($f$, $\theta_{t-1}$, $\mathcal{D}_t$, $\mathcal{D}_{1:t-1}$, $m$, $m^{(h)}$)
10: $\mathcal{U}_{t+1} \leftarrow$ Select($f, \theta_t, \mathcal{U}, b_t$) ▷ Select $b_t$ points from $\mathcal{U}$
11: $\mathcal{D}_{t+1} \leftarrow \{(x_i, y_i) | x_i \in \mathcal{U}_{t+1}\}$
12: $\mathcal{U} \leftarrow \mathcal{U} \setminus \mathcal{U}_{t+1}$; $\mathfrak{L} \leftarrow \mathfrak{L} \cup \mathcal{D}_{t+1}$; $t \leftarrow t + 1$
13: $\theta_T \leftarrow$ ContinualTrain($f$, $\theta_{T-1}$, $\mathcal{D}_T$, $\mathcal{D}_{1:T-1}$, $m$, $m^{(h)}$)
14: return $\mathfrak{L}$, $\theta_T$

**Experience Replay (CAL-ER)** is the simplest and oldest replay-based method (Ratcliff, 1990; Robins, 1995). In this approach, $\mathcal{B}_{\text{current}}$ and $\mathcal{B}_{\text{replay}}$ are interleaved to create a minibatch $\mathcal{B}$ of size $m + m^{(h)}$ and

$\mathcal{B}_{\text{replay}}$ is chosen uniformly at random from $\mathcal{D}_{1:t-1}$. The parameters $\theta$ of model $f$ are updated based on the gradient computed on $\mathcal{B}$.

**Maximally Interfered Retrieval (CAL-MIR)** chooses a size-$m^{(h)}$ subset of points from $\mathcal{D}_{1:t-1}$ most likely to be forgotten (Aljundi et al., 2019a). Given a batch of $m$ labeled samples $\mathcal{B}_{\text{current}}$ sampled from $\mathcal{D}_t$ and model parameters $\theta$, $\theta_v$ is computed by taking a "virtual" gradient step i.e., $\theta_v = \theta - \eta \nabla \mathcal{L}_c(\theta)$ where $\eta$ is the learning rate. Then for every example $x$ in the history, $s_{MIR}(x) = \ell(f(x;\theta), y) - \ell(f(x;\theta_v), y)$ (i.e., the change in loss after taking a single gradient step) is computed. The $m^{(h)}$ samples with the highest $s_{MIR}$ score are selected for $\mathcal{B}_{\text{replay}}$, and the remainder is similar to experience replay. $\mathcal{B}_{\text{current}}$ and $\mathcal{B}_{\text{replay}}$ are concatenated to form the minibatch (as in CAL-ER), upon which the gradient update is computed. In practice, selection is done on a random subset of $\mathcal{D}_{1:t-1}$ for speed, since computing $s_{MIR}$ for every historical sample is prohibitively expensive.

**Dark Experience Replay (CAL-DER)** uses a distillation approach to regularize updates (Buzzega et al., 2020). Let $g(x;\theta)$ denote the pre-softmax logits of classifier $f(x;\theta)$, i.e., $f(x;\theta) = \text{softmax}(g(x;\theta))$. In DER, every $x' \in \mathcal{D}_{1:t-1}$ has an associated $z'$ which corresponds to the model's logits at the end of the task when $x$ was first observed — if $x' \in \mathcal{D}_{t'}$, then $z' \triangleq g(x'; \theta^*_{t'}))$ where $t' \in [t-1]$ and $\theta^*_{t'}$ are the parameters obtained after round $t'$. DER minimizes $\mathcal{L}_{\text{DER}}(\theta)$ defined as:

$$\mathcal{L}_{\text{DER}}(\theta) \triangleq \mathcal{L}_c(\theta) + \mathop{\mathbb{E}}_{(x',y',z') \sim \mathcal{B}_{\text{replay}}} \left[ \alpha \, \|g(x';\theta) - z'\|_2^2 + \beta \, \ell(y', f(x';\theta)) \right], \tag{1}$$

where $\mathcal{B}_{\text{replay}}$ is a batch uniformly at randomly (w/o replacement) sampled from $\mathcal{D}_{1:t-1}$, and $\alpha$ and $\beta$ are tuneable hyperparameters. The first term ensures that samples from the current task are classified correctly. The second term consists of a classification loss and a mean squared error (MSE) based distillation loss applied to historical samples.

**Scaled Distillation (CAL-SD)** $\mathcal{L}_{\text{SD}}(\theta)$ is a new objective proposed in this work defined via:

$$\mathcal{L}_{\text{replay}}(\theta) \triangleq \mathop{\mathbb{E}}_{(x',y',z') \sim \mathcal{B}_{\text{replay}}} \left[ \alpha \, D_{\text{KL}} \left( \text{softmax}(z') \mid\mid f(x';\theta) \right) + (1-\alpha) \, \ell \left( y', f(x';\theta) \right) \right], \tag{2}$$

and then $\mathcal{L}_{\text{SD}}(\theta) \triangleq \lambda_t \, \mathcal{L}_c(\theta) + (1-\lambda_t) \, \mathcal{L}_{\text{replay}}(\theta)$ where $\lambda_t \triangleq |\mathcal{D}_t| / (|\mathcal{D}_t| + |D_{1:t-1}|)$. Similar to CAL-DER, $\mathcal{L}_{\text{replay}}$ is a sum of two terms: a distillation loss and a classification loss. The distillation loss expresses the KL-divergence between the posterior probabilities produced by $f$ and $\text{softmax}(z')$, where $z'$ is defined in the DER section. We use KL-divergence instead of MSE loss on the logits so that the distillation and the classification losses have the same scale and dynamic range. $\alpha \in [0, 1]$ is a tuneable hyperparameter.

The weight of each term is determined adaptively by a "stability/plasticity" trade-off term $\lambda_t$. A stability-plasticity dilemma is commonly found in both biological and artificial neural networks (Abraham & Robins, 2005; Mermillod et al., 2013). A network is *stable* if it can effectively retain past information but cannot adapt to new tasks efficiently, whereas a network that is *plastic* can quickly learn new tasks but is prone to forgetting. The trade-off between stability and plasticity is a well-known constraint in CL (Mermillod et al., 2013). For CAL, we want the model to be plastic early on, and stable later on. We apply this intuition with $\lambda_t$: higher values indicate higher plasticity, since minimizing the classification error of samples from the current task is prioritized. Since $\mathcal{D}_{1:t-1}$ increases with $t$, $\lambda_t$ decreases and the model becomes more stable in later training rounds.

**Scaled Distillation w/ Submodular Sampling (CAL-SDS2)** CAL-SDS2 is another new CL approach we introduce in this work. CAL-SDS2 uses CAL-SD to regularize the model and uses a submodular sampling procedure to select a diverse set of history points to replay. Submodular functions are well-known to be suited to capture notions of diversity and representativeness (Lin & Bilmes, 2011; Wei et al., 2015; Bilmes, 2022) and the simple greedy algorithm can approximately maximize, under a cardinality constraint, a monotone submodular function up to a $1 - e^{-1}$ constant factor multiplicative guarantee (Fisher et al., 1978; Minoux, 1978; Mirzasoleiman et al., 2015). We define our submodular function $G$ as:

$$G(\mathcal{S}) \triangleq \sum_{x_i \in \mathcal{A}} \max_{x_j \in \mathcal{S}} w_{ij} + \lambda \log \left( 1 + \sum_{x_i \in \mathcal{S}} h(x_i) \right). \tag{3}$$

The first term is a facility location function, where $w_{ij}$ is a similarity score between samples $x_i$ and $x_j$. We use $w_{ij} = \exp\left(-\|z_i - z_j\|^2/2\sigma^2\right)$ where $z_i$ is the penultimate layer of model $f$ for $x_i$ and $\sigma$ is a hyperparameter. The second term is a concave over modular function (Liu et al., 2013) and $h(x_i)$ is a standard AL measure of model uncertainty, such as entropy of the model's output distribution. Both terms are well known to be monotone non-decreasing submodular, as is their non-negatively weighted sum. The core reason for applying a concave function over a model-uncertainty-score-based modular function, instead of keeping it as a pure modular function, is to better align the modular uncertainty values with the facility location function. Otherwise, if we do not apply the concave function, the facility location function dominates during the early steps of the greedy algorithm and the modular function dominates in the later steps of greedy. In order to speed up SDS2 (specifically to avoid the need to perform forward passes through the entire history before a single step), we randomly subsample from the history before performing submodular maximization so $\mathcal{S} \subset \mathcal{A} \subset \mathcal{D}_{1:t-1}$. The objective of CAL-SDS2 is to ensure that the set of samples that are replayed are both difficult and diverse, similar to the motivation of the heuristic employed in Wei et al. (2015).

**Baselines** All proposed CAL methods are compared against two AL baselines. The first baseline is standard active learning, denoted as AL, which is no different from the procedure shown in Algorithm 1. We also consider active learning with warm starting (AL w/ WS), which uses the converged model from the previous round to initialize the model for the current round. Both models retrain on the entire dataset at each query round.

## 5 Experiments and Results

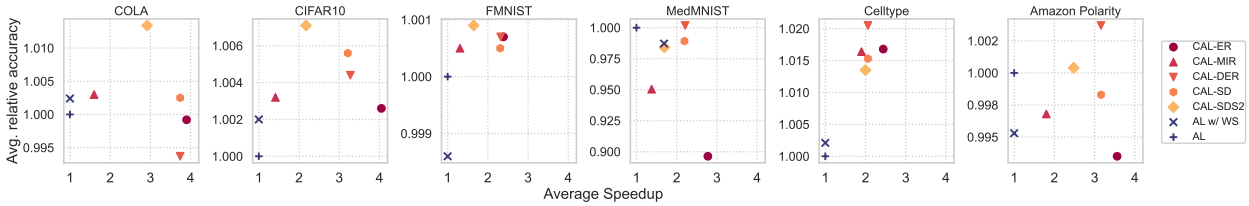


Figure 3: Relative accuracy vs. speedup averaged over different labeling budgets. For every dataset, we have *at least one* CAL method, that is faster with same accuracy than baseline (*top right region is desiderata*).

We evaluate the validation performance of the model when we train on different fractions ($b/n$) of the full dataset. We compute the *speedup* attained by a CAL method by dividing the wall-clock training time of the baseline AL method over the wall-clock training time of the CAL method. We test the CAL methods on a variety of different datasets spanning multiple modalities. Two baselines do not utilize CAL which are standard *AL* (active learning) as well as *AL w/ WS* (active learning with warm starting but still training using all the presently available labeled data).

Our objective is to demonstrate: (1) *at least one* CAL-based method exists that can match or outperform a standard active learning technique while achieving a significant speedup for every budget and dataset and (2) models that have been trained using a CAL-based method behave no differently than standard models. We emphasize that the purpose of this work is *not* to champion a single method, but rather to showcase an assortment of approaches in the CAL paradigm that achieve different performance/speedup trade-offs. Lastly, we would like to point out that some of the CAL methods are ablations of each other. For example, CAL-ER is ablation for CAL-DER (or CAL-SD) when we replace the distillation component. Similarly, CAL-SD is ablation of CAL-SDS2, where we remove the submodular selection part.

### 5.1 Datasets and Experimental Setup

We use the following datasets, which span a spectrum of data modalities, scale (both in terms of dataset size, and model's computational/memory footprint), and class balance.

**FMNIST:** The FMNIST dataset consists of 70,000 28×28 grayscale images of fashion items belonging to 10 classes (Xiao et al., 2017). A ResNet-18 architecture (He et al., 2016) with SGD is used. We apply data augmentation, as in Beck et al. (2021), consisting of random horizontal flips and random croppings.

**CIFAR-10:** CIFAR-10 consists of 60,000 32×32 color images with 10 different categories (Krizhevsky, 2009). We use a ResNet-18 and use the SGD optimizer for all CIFAR-10 experiments. We apply data augmentations consisting of random horizontal flips and random croppings.

**MedMNIST:** We use the DermaMNIST dataset within the MedMNIST collection (Yang et al., 2021a;b) for performance evaluation of CAL on medical imaging modalities. It consists of 3-color channel dermatoscopy images of 7 different skin diseases, originally obtained from Codella et al. (2019); Tschandl et al. (2018). A ResNet-18 architecture is used for all DermaMNIST experiments.

**Amazon Polarity Review:** (Zhang et al., 2015) is an NLP dataset consisting of reviews from Amazon and their corresponding star-ratings (5 classes) which was used for active learning in Coleman et al. (2020b). Similar to the previous work we consider a total unlabeled pool of size 2 million sentences and use a VDCNN-9 (Schwenk et al., 2017) architecture trained using Adam.

**COLA:** COLA (Warstadt et al., 2018) aims to check the linguistic acceptability of a sentence via binary classification. We consider an unlabeled size-7000 pool similar to Ein-Dor et al. (2020) and use a BERT (Devlin et al., 2019) backbone trained using Adam.

**Single-Cell Cell Type Identity Classification:** Recent single-cell RNA sequencing (scRNA-seq) technologies have enabled large-scale characterization of hundreds of thousands to millions of cells in complex tissues, and accurate cell type annotation is a crucial step in the study of such datasets. To this end, several deep learning models have been proposed to automatically label new scRNA-seq datasets (Xie et al., 2021). The HCL dataset is highly *class-imbalanced* and consists of scRNA-seq data for 562,977 cells across 63 cell types represented in 56 human tissues (Han et al., 2020). We use the ACTINN model (Ma & Pellegrini, 2019), a four-layer multi-layer perceptron that predicts the cell-type for each cell given its expression of 28832 genes, and uses an SGD optimizer.

**Hyperparameters:** Details about the specific hardware and the choices of hyperparameters used to train models for each technique can be found in Appendix A.5.

**Active Learning setup:** As done in previous work (Coleman et al., 2020b; Killamsetty et al., 2021a) for CIFAR10 and Amazon polarity review, budgets go from 10% to 50% in increments of 10% (results for other query sizes are presented in Appendix C.3). For FMNIST, MedMNIST, and Cell-type datasets, it goes from 10% to 30% in increments of 5%. Lastly, for COLA, we follow a budget using absolute sizes from 200 to 1000 in increments of 200 (similar to Ein-Dor et al. (2020)).[2] We adopt the AL framework proposed in Beck et al. (2021) for all experiments. In the main paper, we here present results for an uncertainty sampling-based acquisition function. *However, we provide results using other acquisition functions in Appendix C.*

### 5.2 Performance vs Speedup

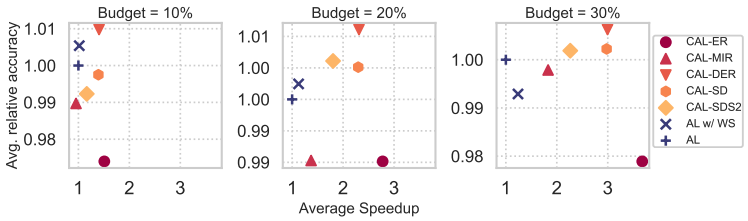


Figure 4: Relative accuracy vs. speedup averaged over different datasets. For every given budget, we have *at least one* CAL method that is faster with the same (if not better) accuracy than the baseline.

Our goal is to show that for every dataset and budget, there exists at least one CAL method that performs equivalent (if not better) than baseline AL. However, with raw accuracies, it is difficult to compare different CAL methods and baseline AL over different datasets at different budgets. Therefore, we begin by observing the relative gain in accuracy over the AL baseline. Relative gains further make it feasible to take an average across the budgets, for a given dataset, and to take averages across the datasets, for a given budget. For *every budget*, we normalize the accuracies of each method by that of baseline AL. This makes the baseline accuracy always **1**, irrespective of budget and dataset. Relative performances greater than 1 indicate better than baseline accuracy (and the opposite for the less than 1 case). Having said that: (1) keeping the budgets fixed to 10%, 20%, and 30% and averaging over the datasets (except COLA, since it has a different budget)

[2]We follow the query set sizes from (Ein-Dor et al., 2020)

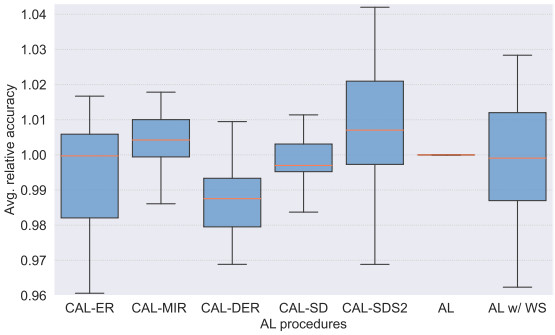


Figure 5: Comparison of CAL methods with the baseline on CIFAR-10C; absolute accuracies on individual benchmark and its average difference with the baseline AL are reported in Appendix A.2 table 7 and 9, respectively. All CAL methods perform within a standard deviation of the standard AL baseline, and CAL-SDS2 achieves the highest robust accuracy on average.

give us Figure 4; (2) keeping the dataset fixed, averaging the relative accuracy vs. speedups across different budgets give us Figure 3; and (3) further averaging the above across different datasets give us Figure 1. Methods in the top right corner are preferable. For space reasons, we display only relative accuracies in the main paper, but all absolute accuracies and standard deviations for each method for every dataset and every budget are available in Appendix A.2.

From the overall and per-dataset depiction of CAL's performance (Figures 1 and 3, respectively), it is evident that *there exists a CAL method that attains a significant speedup over a standard AL technique for every dataset and budget while preserving test set accuracy.* From Figure 3, we can further see that for some datasets (such as FMNIST and CIFAR-10), CAL-ER, a non-distillation, and uniform sampling-based method, only incur a minor drop in performance but attain the highest speedup. This suggests that naively biasing learning towards recent tasks can be sufficient to adapt the model to a new set of points between AL rounds. However, as we show in Figure 4 it is not universally true for all the datasets (at different budgets). Hence, the methods which include some type of distillation term (CAL-DER, CAL-SD, CAL-SDS2) generally perform the best out of all CAL methods. We believe that the submodular sampling-based method (CAL-SDS2) can be accelerated using stochastic methods and results improved by considering other submodular functions, which we leave as future work. It should be mentioned, however, that the concave function on $h(x_i)$ was essential for CAL-SDS2's performance.

### 5.3 Comparison Between Standard and CAL Models

In this section, we next assess whether CAL training has any adverse effect on the final model's behavior. We first demonstrate that CAL does *not* result in any deterioration of model robustness (Section 5.3.1). We then demonstrate that CAL models and baseline trained models are uncertain about a similar set of unseen examples (Section 5.3.2). Lastly, in appendix A.5 we provide a sensitivity analysis of our proposed methods, where we demonstrate that *CAL methods are robust to the changes to the hyperparameters.*

#### 5.3.1 Robustness

Test time distributions can vary from training distributions, so it is important to ensure that models can generalize across different domains. Since models trained using CAL methods require significantly fewer gradient steps, the modified training procedure may produce fickler models that are less robust to domain shifts. To ensure against this, we evaluate CAL-method-trained model robustness in this section. We consider CIFAR-10C Hendrycks & Dietterich (2019), a dataset comprising 19 different corruptions each done at 5 levels of severity. For each model trained up to a 50% budget, we report the average classification accuracy over each corruption and compare it against the baseline in Figure 5; each result is an average of over three random seeds. We note that most of the CAL methods perform statistically similarly to standard active learning, all while providing significant acceleration. Moreover, on average across all the tests, **models trained with CAL-SDS2 are better than the models trained with CAL-DER**, where we see a difference of as much as 5% in corruptions such as *glass blur*; please refer to appendix A.3 table 7, table 8 and 9 for the absolute accuracies and statistical tests. Submodular sampling replays a diverse representative subset of history which is likely the reason behind CAL-SDS2's better robustness. The relationship of diversity with robustness has also been explored in previous works including Killamsetty et al. (2021b); Fang et al. (2022); Rozen et al. (2019); Gong et al. (2018).

### 5.3.2 Correlation of Uncertainty Scores

For models trained using CAL techniques to be used as valid substitutes for standard AL models, these two classes of models need to query similar samples at each AL round. This is particularly important if AL is being used solely as a data subset selection procedure (where the user is concerned about the quality of the resulting labeled dataset as opposed to the final model). When using uncertainty sampling as the AL acquisition function, computing the Pearson correlation between the entropy scores of baseline and CAL models on the validation set after every query round is one way of determining this. Since most AL policies incorporate some notion of uncertainty, we hypothesize that the results of this experiment should extend to other acquisition functions as well. A similar analysis is done in Coleman et al. (2020b).

As seen in Figure 6, the Pearson correlation between all pairs of models is positive at every AL query round. Thus, the nature of samples that the models are uncertain about, and thus are likely to be chosen at each round, is similar between the CAL-trained models and baseline AL-trained models. A breakdown of these correlations at every round is provided in Figure 13 in the Appendix.

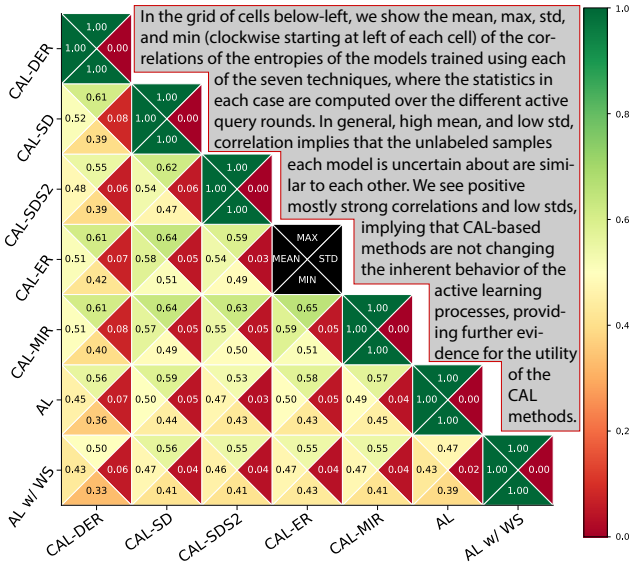


Figure 6: Cross-Method Entropy Correlation Statistics.

## 6 Conclusion and Future Work

We proposed the CAL framework, the first method to circumvent the problem of having to retrain models between batch AL rounds. Across vision, natural language, medical imaging, and biological datasets, we show there is always a CAL-based method that either matches or outperforms standard AL while achieving considerable speedups. Since CAL is independent of the model architecture and AL strategy, this framework applies to a broad range of settings. Future research directions may include the following: **(1):** CAL reduces the training time of the model, but not the AL query time although query time reductions could be an offshoot of this work; **(2):** CAL operates using existing AL query acquisition functions, but it is possible to tailor acquisition functions for CAL methods yielding further generalization and computational improvements; **(3):** while SDS2's use of submodularity helped robustness, additional submodular strategies can be used to further improve results and also for diverse AL query selection; and **(4):** CAL provides a novel application for CL; future CL work can be partially assessed based on its CAL performance.

## References

Wickliffe C Abraham and Anthony Robins. Memory retention–the synaptic stability versus plasticity dilemma. *Trends Neurosci*, 28(2):73–78, February 2005.

Rahaf Aljundi, Lucas Caccia, Eugene Belilovsky, Massimo Caccia, Min Lin, Laurent Charlin, and Tinne Tuytelaars. Online continual learning with maximally interfered retrieval. *CoRR*, abs/1908.04742, 2019a. URL `http://arxiv.org/abs/1908.04742`.

Rahaf Aljundi, Min Lin, Baptiste Goujaud, and Yoshua Bengio. Gradient based sample selection for online continual learning. *arXiv preprint arXiv:1903.08671*, 2019b.

Jordan Ash and Ryan P Adams. On warm-starting neural network training. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 3884–3894. Curran Associates, Inc., 2020. URL `https://proceedings.neurips.cc/paper/2020/file/288cd2567953f06e460a33951f55daaf-Paper.pdf`.

Jordan T. Ash, Chicheng Zhang, Akshay Krishnamurthy, John Langford, and Alekh Agarwal. Deep batch active learning by diverse, uncertain gradient lower bounds. *ArXiv*, abs/1906.03671, 2020.

Ali Ayub and Carter Fendley. Few-shot continual active learning by a robot, 2022. URL https://arxiv.org/abs/2210.04137.

Maria-Florina Balcan, Steve Hanneke, and Jennifer Wortman Vaughan. The true sample complexity of active learning. *Machine Learning*, 80(2-3):111–139, April 2010. doi: 10.1007/s10994-010-5174-y. URL https://doi.org/10.1007/s10994-010-5174-y.

Nathan Beck, Durga Sivasubramanian, Apurva Dani, Ganesh Ramakrishnan, and Rishabh K. Iyer. Effective evaluation of deep active learning on image classification tasks. *CoRR*, abs/2106.15324, 2021. URL https://arxiv.org/abs/2106.15324.

Emily M. Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. On the dangers of stochastic parrots: Can language models be too big? In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, FAccT '21, pp. 610–623, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450383097. doi: 10.1145/3442188.3445922. URL https://doi.org/10.1145/3442188.3445922.

Jeff A. Bilmes. Submodularity in machine learning and artificial intelligence. *CoRR*, abs/2202.00132, 2022. URL https://arxiv.org/abs/2202.00132.

Pietro Buzzega, Matteo Boschini, Angelo Porrello, Davide Abati, and SIMONE CALDERARA. Dark experience for general continual learning: a strong, simple baseline. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 15920–15930. Curran Associates, Inc., 2020.

Arslan Chaudhry, Marc' Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny. Efficient lifelong learning with a-gem. In *ICLR*, 2019.

Arslan Chaudhry, Albert Gordo, David Lopez-Paz, Puneet K. Dokania, and Philip Torr. Using hindsight to anchor past knowledge in continual learning, 2020. URL https://openreview.net/forum?id=Hke12T4KPS.

Noel Codella, Veronica Rotemberg, Philipp Tschandl, M. Emre Celebi, Stephen Dusza, David Gutman, Brian Helba, Aadi Kalloo, Konstantinos Liopyris, Michael Marchetti, Harald Kittler, and Allan Halpern. Skin lesion analysis toward melanoma detection 2018: A challenge hosted by the international skin imaging collaboration (isic), 2019. URL https://arxiv.org/abs/1902.03368.

David A. Cohn, Les E. Atlas, and Richard E. Ladner. Improving generalization with active learning. *Mach. Learn.*, 15(2):201–221, 1994. URL http://dblp.uni-trier.de/db/journals/ml/ml15.html#CohnAL94.

Cody Coleman, Edward Chou, Sean Culatana, Peter Bailis, Alexander C. Berg, Roshan Sumbaly, Matei Zaharia, and I. Zeki Yalniz. Similarity search for efficient active learning and search of rare concepts. *CoRR*, abs/2007.00077, 2020a. URL https://arxiv.org/abs/2007.00077.

Cody Coleman, Christopher Yeh, Stephen Mussmann, Baharan Mirzasoleiman, Peter Bailis, Percy Liang, Jure Leskovec, and Matei Zaharia. Selection via proxy: Efficient data selection for deep learning. In *International Conference on Learning Representations*, 2020b. URL https://openreview.net/forum?id=HJg2b0VYDr.

Ido Dagan and Sean P. Engelson. Committee-based sampling for training probabilistic classifiers. In *Proceedings of the Twelfth International Conference on International Conference on Machine Learning*, ICML'95, pp. 150–157, San Francisco, CA, USA, 1995. Morgan Kaufmann Publishers Inc. ISBN 1558603778.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *ArXiv*, abs/1810.04805, 2019.

Payal Dhar. The carbon impact of artificial intelligence. *Nature Machine Intelligence*, 2(8):423–425, Aug 2020. ISSN 2522-5839. doi: 10.1038/s42256-020-0219-9. URL https://doi.org/10.1038/s42256-020-0219-9.

Liat Ein-Dor, Alon Halfon, Ariel Gera, Eyal Shnarch, Lena Dankin, Leshem Choshen, Marina Danilevsky, Ranit Aharonov, Yoav Katz, and Noam Slonim. Active Learning for BERT: An Empirical Study. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 7949–7962, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.638. URL https://aclanthology.org/2020.emnlp-main.638.

Seyda Ertekin, Jian Huang, Leon Bottou, and Lee Giles. Learning on the border: Active learning in imbalanced data classification. In *Proceedings of the Sixteenth ACM Conference on Conference on Information and Knowledge Management*, CIKM '07, pp. 127–136, New York, NY, USA, 2007. Association for Computing Machinery. ISBN 9781595938039. doi: 10.1145/1321440.1321461. URL https://doi.org/10.1145/1321440.1321461.

Alex Fang, Gabriel Ilharco, Mitchell Wortsman, Yuhao Wan, Vaishaal Shankar, Achal Dave, and Ludwig Schmidt. Data determines distributional robustness in contrastive language image pre-training (clip), 2022.

Sebastian Farquhar and Yarin Gal. Towards robust evaluations of continual learning. *arXiv preprint arXiv:1805.09733*, 2018.

M.L. Fisher, G.L. Nemhauser, and L.A. Wolsey. An analysis of approximations for maximizing submodular set functions—II. In *Polyhedral combinatorics*, 1978.

Robert M French. Catastrophic forgetting in connectionist networks. *Trends in cognitive sciences*, 3(4): 128–135, 1999.

Zhiqiang Gong, Ping Zhong, and Weidong Hu. Diversity in machine learning. *CoRR*, abs/1807.01477, 2018. URL http://arxiv.org/abs/1807.01477.

Xiaoping Han, Ziming Zhou, Lijiang Fei, Huiyu Sun, Renying Wang, Yao Chen, Haide Chen, Jingjing Wang, Huanna Tang, Wenhao Ge, Yincong Zhou, Fang Ye, Mengmeng Jiang, Junqing Wu, Yanyu Xiao, Xiaoning Jia, Tingyue Zhang, Xiaojie Ma, Qi Zhang, Xueli Bai, Shujing Lai, Chengxuan Yu, Lijun Zhu, Rui Lin, Yuchi Gao, Min Wang, Yiqing Wu, Jianming Zhang, Renya Zhan, Saiyong Zhu, Hailan Hu, Changchun Wang, Ming Chen, He Huang, Tingbo Liang, Jianghua Chen, Weilin Wang, Dan Zhang, and Guoji Guo. Construction of a human cell landscape at single-cell level. *Nature*, 581(7808):303–309, March 2020.

Steve Hanneke. A bound on the label complexity of agnostic active learning. In *Proceedings of the 24th International Conference on Machine Learning*, ICML '07, pp. 353–360, New York, NY, USA, 2007. Association for Computing Machinery. ISBN 9781595937933. doi: 10.1145/1273496.1273541. URL https://doi.org/10.1145/1273496.1273541.

Steve Hanneke. *Theoretical foundations of active learning*. Carnegie Mellon University, 2009.

Steve Hanneke. Activized learning: Transforming passive to active with improved label complexity. *Journal of Machine Learning Research*, 13(49):1469–1587, 2012. URL http://jmlr.org/papers/v13/hanneke12a.html.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2016. doi: 10.1109/CVPR.2016.90.

Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. *Proceedings of the International Conference on Learning Representations*, 2019.

Zhihao Jia, Colin Unger, Wei Wu, Sina Lin, Mandeep Baines, Vinay Ramakrishnaiah Carlos Efrain, Nirmal Prajapati, Pat McCormick, Jamaludin Mohd-Yusof, Xi Luo, Dheevatsa Mudigere, Jongsoo Park, Misha Smelyanskiy, and Alex Aiken. Unity: Accelerating DNN training through joint optimization of algebraic transformations and parallelization. In *16th USENIX Symposium on Operating Systems Design and Implementation (OSDI 22)*, Carlsbad, CA, July 2022. USENIX Association. URL https://www.usenix.org/conference/osdi22/presentation/jia.

KrishnaTeja Killamsetty, Durga Sivasubramanian, Ganesh Ramakrishnan, and Rishabh K. Iyer. GLISTER: generalization based data subset selection for efficient and robust learning. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pp. 8110–8118. AAAI Press, 2021a. URL https://ojs.aaai.org/index.php/AAAI/article/view/16988.

KrishnaTeja Killamsetty, Xujiang Zhao, Feng Chen, and Rishabh K. Iyer. RETRIEVE: coreset selection for efficient and robust semi-supervised learning. *CoRR*, abs/2106.07760, 2021b. URL https://arxiv.org/abs/2106.07760.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526, 2017a. doi: 10.1073/pnas.1611835114. URL https://www.pnas.org/doi/abs/10.1073/pnas.1611835114.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526, 2017b. doi: 10.1073/pnas.1611835114.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017c.

Andreas Kirsch, Joost van Amersfoort, and Yarin Gal. Batchbald: Efficient and diverse batch acquisition for deep bayesian active learning. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper/2019/file/95323660ed2124450caaac2c46b5ed90-Paper.pdf.

Alex Krizhevsky. Learning multiple layers of features from tiny images. pp. 32–33, 2009. URL https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf.

David D. Lewis and Jason Catlett. Heterogeneous uncertainty sampling for supervised learning. In William W. Cohen and Haym Hirsh (eds.), *Machine Learning Proceedings 1994*, pp. 148–156. Morgan Kaufmann, San Francisco (CA), 1994. ISBN 978-1-55860-335-6. doi: https://doi.org/10.1016/B978-1-55860-335-6.50026-X. URL https://www.sciencedirect.com/science/article/pii/B978155860335650026X.

David D. Lewis and William A. Gale. A sequential algorithm for training text classifiers, 1994. URL http://arxiv.org/abs/cmp-lg/9407020. active learning roots.

Haoran Li, Aditya Krishnan, Jingfeng Wu, Soheil Kolouri, Praveen K. Pilly, and Vladimir Braverman. Lifelong learning with sketched structural regularization. *CoRR*, abs/2104.08604, 2021. URL https://arxiv.org/abs/2104.08604.

Tian Li, Anit Kumar Sahu, Ameet Talwalkar, and Virginia Smith. Federated learning: Challenges, methods, and future directions. *IEEE signal processing magazine*, 37(3):50–60, 2020.

Xiao Li and Jeff Bilmes. A Bayesian divergence prior for classifier adaptation. In *Artificial Intelligence and Statistics*, pp. 275–282. PMLR, 2007.

Hui Lin and Jeff Bilmes. A class of submodular functions for document summarization. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pp. 510–520, Portland, Oregon, USA, June 2011. Association for Computational Linguistics. URL https://aclanthology.org/P11-1052.

Yuzong Liu, Kai Wei, Katrin Kirchhoff, Yisong Song, and Jeff Bilmes. Submodular feature selection for high-dimensional acoustic score spaces. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 7184–7188, 2013. doi: 10.1109/ICASSP.2013.6639057.

David Lopez-Paz and Marc' Aurelio Ranzato. Gradient episodic memory for continual learning. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

Feiyang Ma and Matteo Pellegrini. ACTINN: automated identification of cell types in single cell RNA sequencing. *Bioinformatics*, 36(2):533–538, 07 2019. ISSN 1367-4803. doi: 10.1093/bioinformatics/btz592. URL https://doi.org/10.1093/bioinformatics/btz592.

C. Mayer and R. Timofte. Adversarial sampling for active learning. In *2020 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 3060–3068, Los Alamitos, CA, USA, mar 2020. IEEE Computer Society. doi: 10.1109/WACV45572.2020.9093556. URL https://doi.ieeecomputersociety.org/10.1109/WACV45572.2020.9093556.

James L McClelland, Bruce L McNaughton, and Randall C O'Reilly. Why there are complementary learning systems in the hippocampus and neocortex: insights from the successes and failures of connectionist models of learning and memory. *Psychological review*, 102(3):419, 1995.

Michael McCloskey and Neal J Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, pp. 109–165. Elsevier, 1989.

Martial Mermillod, Aurélia Bugaiska, and Patrick BONIN. The stability-plasticity dilemma: investigating the continuum from catastrophic forgetting to age-limited learning effects. *Frontiers in Psychology*, 4, 2013. ISSN 1664-1078. doi: 10.3389/fpsyg.2013.00504. URL https://www.frontiersin.org/article/10.3389/fpsyg.2013.00504.

M. Minoux. Accelerated greedy algorithms for maximizing submodular set functions. In *Optimization Techniques*, 1978.

Baharan Mirzasoleiman, Ashwinkumar Badanidiyuru, Amin Karbasi, Jan Vondrák, and Andreas Krause. Lazier than lazy greedy. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 29, 2015.

Matthias Perkonigg, Johannes Hofmanninger, and Georg Langs. Continual active learning for efficient adaptation of machine learning models to changing image acquisition. In *Lecture Notes in Computer Science*, Lecture notes in computer science, pp. 649–660. Springer International Publishing, Cham, 2021.

Robert Pinsler, Jonathan Gordon, Eric Nalisnick, and José Miguel Hernández-Lobato. Bayesian batch active learning as sparse subset approximation. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper/2019/file/84c2d4860a0fc27bcf854c444fb8b400-Paper.pdf.

R Ratcliff. Connectionist models of recognition memory: constraints imposed by learning and forgetting functions. *Psychol Rev*, 97(2):285–308, April 1990.

Matthew Riemer, Ignacio Cases, Robert Ajemian, Miao Liu, Irina Rish, Yuhai Tu, and Gerald Tesauro. Learning to learn without forgetting by maximizing transfer and minimizing interference. *CoRR*, abs/1810.11910, 2018.

Anthony Robins. Catastrophic forgetting, rehearsal and pseudorehearsal. *Connection Science*, 7(2):123–146, 1995. doi: 10.1080/09540099550039318. URL https://doi.org/10.1080/09540099550039318.

Ohad Rozen, Vered Shwartz, Roee Aharoni, and Ido Dagan. Diversify your datasets: Analyzing generalization via controlled variance in adversarial datasets. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pp. 196–205, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/K19-1019. URL https://aclanthology.org/K19-1019.

Roy Schwartz, Jesse Dodge, Noah A. Smith, and Oren Etzioni. Green ai. *Commun. ACM*, 63(12):54–63, nov 2020. ISSN 0001-0782. doi: 10.1145/3381831. URL https://doi.org/10.1145/3381831.

Holger Schwenk, Loïc Barrault, Alexis Conneau, and Yann LeCun. Very deep convolutional networks for text classification. In *EACL*, 2017.

Ozan Sener and Silvio Savarese. Active learning for convolutional neural networks: A core-set approach, 2017. URL https://arxiv.org/abs/1708.00489.

Ozan Sener and Silvio Savarese. Active learning for convolutional neural networks: A core-set approach. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=H1aIuk-RW.

Yuya Senzaki and Christian Hamelain. Active learning for deep neural networks on edge devices, 2021. URL https://arxiv.org/abs/2106.10836.

Burr Settles. Active learning. Morgan & Claypool Publishers, 2012.

Burr Settles. Active learning literature survey. Computer Sciences Technical Report 1648, University of Wisconsin–Madison, 2009. URL http://axon.cs.byu.edu/~martinez/classes/778/Papers/settles.activelearning.pdf.

Burr Settles. From theories to queries: Active learning in practice. In Isabelle Guyon, Gavin Cawley, Gideon Dror, Vincent Lemaire, and Alexander Statnikov (eds.), *Active Learning and Experimental Design workshop In conjunction with AISTATS 2010*, volume 16 of *Proceedings of Machine Learning Research*, pp. 1–18, Sardinia, Italy, 16 May 2011. PMLR. URL https://proceedings.mlr.press/v16/settles11a.html.

Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. Megatron-lm: Training multi-billion parameter language models using model parallelism. *CoRR*, abs/1909.08053, 2019. URL http://arxiv.org/abs/1909.08053.

Michalis K. Titsias, Jonathan Schwarz, Alexander G. de G. Matthews, Razvan Pascanu, and Yee Whye Teh. Functional regularisation for continual learning with gaussian processes. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=HkxCzeHFDB.

Philipp Tschandl, Cliff Rosendahl, and Harald Kittler. The ham10000 dataset, a large collection of multi-source dermatoscopic images of common pigmented skin lesions. *Scientific Data*, 5, 2018.

Matthew Wallingford, Aditya Kusupati, Keivan Alizadeh-Vahid, Aaron Walsman, Aniruddha Kembhavi, and Ali Farhadi. FLUID: A unified evaluation framework for flexible sequential data. *Transactions on Machine Learning Research*, 2023. ISSN 2835-8856. URL https://openreview.net/forum?id=UvJBKWaSSH.

Alex Warstadt, Amanpreet Singh, and Samuel R Bowman. Neural network acceptability judgments. *arXiv preprint arXiv:1805.12471*, 2018.

Kai Wei, Rishabh Iyer, and Jeff Bilmes. Submodularity in data subset selection and active learning. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*, ICML'15, pp. 1954–1963. JMLR.org, 2015.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 38–45, Online, October 2020. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/2020.emnlp-demos.6.

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *CoRR*, abs/1708.07747, 2017. URL http://arxiv.org/abs/1708.07747.

Bingbing Xie, Qin Jiang, Antonio Mora, and Xuri Li. Automatic cell type identification methods for single-cell RNA sequencing. *Computational and Structural Biotechnology Journal*, 19:5874–5887, 2021. ISSN 2001-0370. doi: https://doi.org/10.1016/j.csbj.2021.10.027. URL https://www.sciencedirect.com/science/article/pii/S2001037021004499.

Jiancheng Yang, Rui Shi, and Bingbing Ni. Medmnist classification decathlon: A lightweight automl benchmark for medical image analysis. In *IEEE 18th International Symposium on Biomedical Imaging (ISBI)*, pp. 191–195, 2021a.

Jiancheng Yang, Rui Shi, Donglai Wei, Zequan Liu, Lin Zhao, Bilian Ke, Hanspeter Pfister, and Bingbing Ni. Medmnist v2: A large-scale lightweight benchmark for 2d and 3d biomedical image classification. *arXiv preprint arXiv:2110.14795*, 2021b.

Donggeun Yoo and In So Kweon. Learning loss for active learning. *CoRR*, abs/1905.03677, 2019. URL http://arxiv.org/abs/1905.03677.

Haoyu Zhang, Logan Stafman, Andrew Or, and Michael J. Freedman. Slaq: Quality-driven scheduling for distributed machine learning. SoCC '17, pp. 390–404, New York, NY, USA, 2017. Association for Computing Machinery. ISBN 9781450350280. doi: 10.1145/3127479.3127490. URL https://doi.org/10.1145/3127479.3127490.

Xiang Zhang, Junbo Jake Zhao, and Yann LeCun. Character-level convolutional networks for text classification. *CoRR*, abs/1509.01626, 2015. URL http://arxiv.org/abs/1509.01626.

Lianmin Zheng, Zhuohan Li, Hao Zhang, Yonghao Zhuang, Zhifeng Chen, Yanping Huang, Yida Wang, Yuanzhong Xu, Danyang Zhuo, Joseph E. Gonzalez, and Ion Stoica. Alpa: Automating inter- and intra-operator parallelism for distributed deep learning. *CoRR*, abs/2201.12023, 2022. URL https://arxiv.org/abs/2201.12023.

Zongwei Zhou, Jae Y. Shin, Suryakanth R. Gurudu, Michael B. Gotway, and Jianming Liang. Active, continual fine tuning of convolutional neural networks for reducing annotation efforts. *Medical Image Analysis*, 71:101997, 2021. ISSN 1361-8415. doi: https://doi.org/10.1016/j.media.2021.101997. URL https://www.sciencedirect.com/science/article/pii/S1361841521000438.

Jia-Jie Zhu and José Bento. Generative adversarial active learning. *CoRR*, abs/1702.07956, 2017. URL http://arxiv.org/abs/1702.07956.

# A Additional Experimental Details on Main Results

## A.1 Procedure for normalized accuracy plots

Here we again explain the procedure to generate the normalized accuracy versus speedup plots, as reported in figures 1, 4, and 3. These plots help us to understand and compare the performance of different CAL methods against the baseline AL. For every dataset, we first get the respective accuracies and speedups of CAL methods and baseline AL, for every budget. This is also reported as a tabular form besides the scatter plot visualizations in section A.2.

For **every budget**, we normalize the accuracies of each method by that of baseline AL. This makes the baseline always at **1**, irrespective of budget and dataset. Relative performances greater than 1 indicate better than baseline accuracy (and similarly for the less than 1 case). Note that, for CIFAR10 and Amazon polarity review, budgets go from 10% to 50% in increments of 10%. However, for FMNIST, MedMNIST, and Celltype datasets, it goes from 10% to 30% in increments of 5%. Lastly for COLA, the budgets are different from the above, therefore, we don't include that when we average across the dataset at a fixed budget. Having said that,

- Keeping the budgets fixed to 10%, 20%, and 30% and averaging over the datasets (except COLA, since it has a different budget) will give us figure 4.
- Keeping the dataset fixed, averaging the relative accuracy v.s. speedups across different budgets will give us figure 3.
- Further averaging the above across different datasets will give us the main result figure 1.

From the results above, we can infer that CAL-DER and specialized methods such as CAL-SD/SDS2 always provide a speedup, all while preserving the accuracy, if not better.

## A.2 Results in Tabular Form

In this section, we expand our results mentioned in section 5.2. In particular, we report the absolute accuracies for each dataset, at every budget, and plot it against the observed speedup. All methods highlighted in blue are methods that use CAL. Note that all the results in this section are for uncertainty-based query pool acquisition functions. The choice of budget scale is taken from the previous works (Beck et al., 2021; Ein-Dor et al., 2020). Each dataset has a different complexity (and we train on them using different neural architectures), therefore they differ in the labeling budget.

**FMNIST** Please refer to table 1 and figure 7.

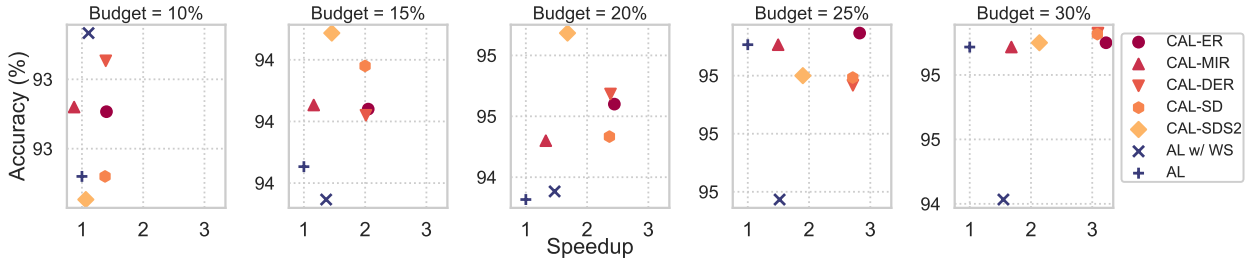


Figure 7: FMNIST Results

| | Test Accuracy (%) | | | | | Factor Speedup | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| **Method** | **10%** | **15%** | **20%** | **25%** | **30%** | **10%** | **15%** | **20%** | **25%** | **30%** |
| CAL-ER | 92.6 ± 0.1 | 93.9 ± 0.2 | 94.5 ± 0.1 | **94.9** ± 0.2 | **94.9** ± 0.2 | 1.5× | 1.4× | 2.0× | 2.4× | 2.8 × |
| CAL-MIR | 92.6 ± 0.3 | 93.9 ± 0.2 | 94.5 ± 0.0 | **94.9** ± 0.1 | **94.9** ± 0.0 | 0.9 × | 1.2× | 1.3× | 1.5× | 1.7× |
| CAL-DER | **92.7** ± 0.1 | 93.9 ± 0.1 | 94.5 ± 0.1 | 94.8 ± 0.2 | **94.9** ± 0.1 | 1.4 × | 2.0× | 2.4× | 2.7× | 3.1× |
| CAL-SD | 92.6 ± 0.1 | **94.0** ± 0.2 | 94.5 ± 0.1 | 94.8 ± 0.2 | **94.9** ± 0.1 | 1.4 × | 2.0× | 2.4× | 2.7× | 3.1× |
| CAL-SDS2 | 92.6 ± 0.1 | **94.0** ± 0.2 | **94.6** ± 0.2 | **94.9** ± 0.1 | **94.9** ± 0.1 | 1.1× | 1.5× | 1.7× | 1.9× | 2.1× |
| AL w/ WS | **92.7** ± 0.3 | 93.8 ± 0.2 | 94.4 ± 0.1 | 94.6 ± 0.1 | 94.4 ± 0.2 | 1.1× | 1.4× | 1.5× | 1.5× | 1.5× |
| AL | 92.6 ± 0.3 | 93.8 ± 0.0 | 94.4 ± 0.1 | **94.9** ± 0.2 | **94.9** ± 0.1 | 1.0× | 1.0× | 1.0× | 1.0× | 1.0× |

Table 1: FMNIST Results

**CIFAR10** Please refer to table 2 and figure 8.

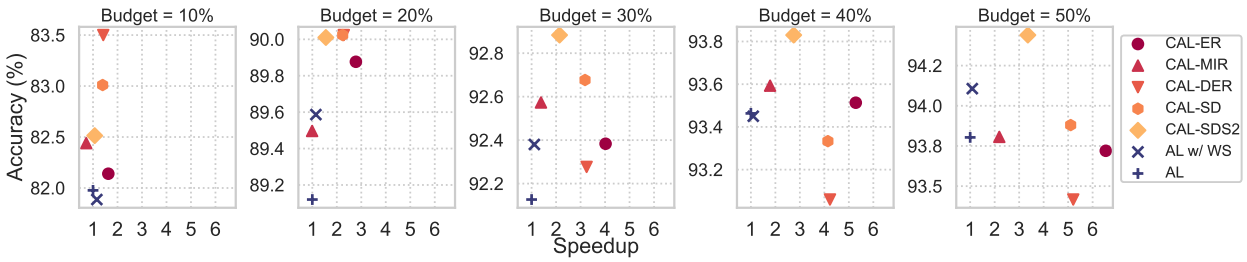


Figure 8: CIFAR-10 Results

| | Test Accuracy (%) | | | | | Factor Speedup | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| **Method** | **10%** | **20%** | **30%** | **40%** | **50%** | **10%** | **20%** | **30%** | **40%** | **50%** |
| CAL-ER | 82.1 ± 0.5 | 89.9 ± 0.3 | 92.4 ± 0.1 | 93.5 ± 0.1 | 93.7 ± 0.3 | 1.6× | 2.8× | 4.0× | 5.3× | 6.5× |
| CAL-MIR | 82.4 ± 0.4 | 89.5 ± 0.3 | 92.6 ± 0.3 | 93.6 ± 0.1 | 93.8 ± 0.2 | 0.7 × | 1.0× | 1.4× | 1.8× | 2.2× |
| CAL-DER | **83.5** ± 0.1 | 90.0 ± 0.4 | 92.3 ± 0.1 | 93.1 ± 0.2 | 93.4 ± 0.1 | 1.4× | 2.3× | 3.2× | 4.2× | 5.2× |
| CAL-SD | 83.0 ± 0.0 | 90.0 ± 0.4 | 92.7 ± 0.2 | 93.3 ± 0.3 | 93.9 ± 0.3 | 1.4× | 2.2× | 3.2× | 4.1× | 5.1× |
| CAL-SDS2 | 82.5 ± 0.1 | **90.1** ± 0.2 | **92.9** ± 0.4 | **94.0** ± 0.2 | **94.4** ± 0.1 | 1.1× | 1.6× | 2.1× | 2.7× | 3.4× |
| AL w/ WS | 81.9 ± 0.4 | 89.6 ± 0.5 | 92.4 ± 0.2 | 93.5 ± 0.1 | 94.1 ± 0.1 | 1.2× | 1.1× | 1.1× | 1.1× | 1.1× |
| AL | 82.0 ± 0.3 | 89.1 ± 0.2 | 92.1 ± 0.4 | 93.5 ± 0.3 | 93.8 ± 0.2 | 1.0× | 1.0× | 1.0× | 1.0× | 1.0× |

Table 2: CIFAR-10 Results

**MedMNIST** Please refer to table 3 and figure 9.

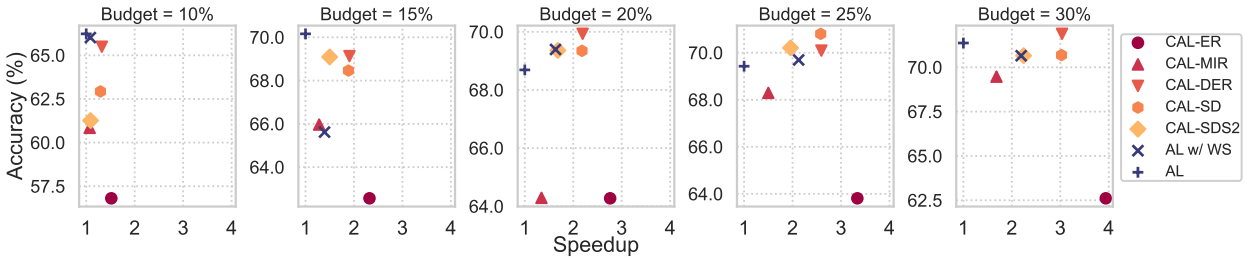


Figure 9: MedMNIST Results

| | Test Accuracy (%) | | | | | Factor Speedup | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| **Method** | **10%** | **15%** | **20%** | **25%** | **30%** | **10%** | **15%** | **20%** | **25%** | **30%** |
| CAL-ER | 64.7 ± 4.4 | 63.1 ± 5.2 | 68.4 ± 1.2 | 66.1 ± 3.7 | 67.5 ± 2.2 | 1.4× | 2.0× | 2.4× | 2.9× | 3.4× |
| CAL-MIR | 64.1 ± 2.8 | 66.2 ± 1.0 | 69.2 ± 1.7 | 70.7 ± 1.1 | 71.8 ± 0.8 | 1.0× | 1.2× | 1.4× | 1.6× | 1.9× |
| CAL-DER | 67.1 ± 2.9 | 67.9 ± 2.0 | **69.8** ± 1.2 | 71.5 ± 0.6 | 72.2 ± 0.6 | 1.2× | 1.6× | 2.1× | 2.6× | 3.2× |
| CAL-SD | 65.1 ± 3.1 | 67.8 ± 2.3 | 70.6 ± 0.9 | 71.1 ± 1.1 | 72.1 ± 1.1 | 1.2× | 1.7× | 2.0× | 2.5× | 2.9× |
| CAL-SDS2 | 66.1 ± 3.6 | 68.1 ± 3.0 | **69.8** ± 1.1 | **71.6** ± 1.6 | **72.5** ± 1.6 | 1.1× | 1.5× | 1.7× | 2.0× | 2.2× |
| AL w/ WS | 67.0 ± 1.4 | 67.5 ± 0.7 | 69.5 ± 0.6 | 70.3 ± 1.0 | 71.3 ± 1.0 | 1.1× | 1.3× | 1.5× | 1.7× | 2.0× |
| AL | **67.2** ± 0.7 | **68.3** ±1.3 | 68.8 ± 0.8 | 69.0 ± 2.0 | 70.9 ± 1.5 | 1.0× | 1.0× | 1.0× | 1.0× | 1.0× |

Table 3: MedMNIST Results

**Amazon Polarity Review** Please refer to table 4 and figure 10.

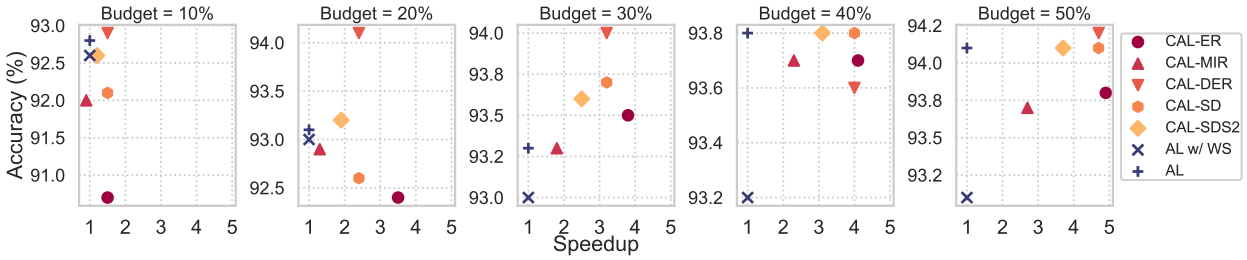


Figure 10: Amazon Polarity Results

| | Test Accuracy (%) | | | | | Factor Speedup | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| **Method** | **10%** | **20%** | **30%** | **40%** | **50%** | **10%** | **20%** | **30%** | **40%** | **50%** |
| CAL-ER | 90.7 ± 3.1 | 92.4 ± 1.2 | 93.5 ± 0.1 | 93.7 ± 0.2 | 93.8 ± 0.2 | 1.5x | 3.5x | 3.3x | 4.1x | 4.9x |
| CAL-MIR | 92.0 ± 0.9 | 92.9 ± 0.1 | 93.3 ± 0.3 | 93.7 ± 0.1 | 93.7 ± 0.2 | 0.9x | 1.3x | 1.8x | 2.3x | 2.7x |
| CAL-DER | **92.9** ± 0.3 | **94.1** ± 0.3 | **94.0** ± 0.7 | 93.6 ± 0.8 | **94.2** ± 0.3 | 1.5x | 2.4x | 3.2x | 4.0x | 4.7x |
| CAL-SD | 92.1 ± 0.3 | 92.6 ± 0.4 | 93.7 ± 0.1 | **93.8** ± 0.1 | 94.1 ± 0.1 | 1.5x | 2.4x | 3.2x | 4.0x | 4.7x |
| CAL-SDS2 | 92.6 ± 0.3 | 93.2 ± 0.1 | 93.6 ± 0.1 | **93.8** ± 0.4 | 94.1 ± 0.0 | 1.2x | 1.9x | 2.5x | 3.1x | 3.7x |
| AL w/ WS | 92.6 ± 0.5 | 93.0 ± 0.2 | 93.0 ± 0.1 | 93.2 ± 0.3 | 93.1 ± 0.1 | 1.0x | 1.0x | 1.0x | 1.0x | 1.0x |
| AL | 92.8 ± 0.2 | 93.1 ± 0.7 | 93.3 ± 1.1 | **93.8** ± 0.5 | 94.1 ± 0.2 | 1.0x | 1.0x | 1.0x | 1.0x | 1.0x |

Table 4: Amazon Polarity Results

**COLA** Please refer to table 5 and figure 11.

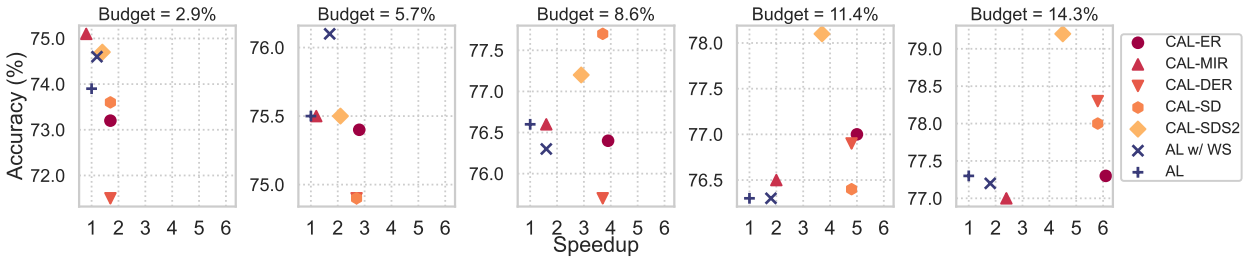


Figure 11: COLA Results

| | Test Accuracy (%) | | | | | Factor Speedup | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| **Method** | **2.9%** | **5.7%** | **8.6%** | **11.4%** | **14.3%** | **2.9%** | **5.7%** | **8.6%** | **11.4%** | **14.3%** |
| CAL-ER | 73.2 ± 1.7 | 75.4 ± 0.8 | 76.4 ± 1.0 | 77.0 ± 2.0 | 77.3 ± 1.4 | 1.7x | 2.8x | 3.9x | 5.0x | 6.1x |
| CAL-MIR | **75.1** ± 0.2 | 75.5 ± 1.2 | 76.6 ± 1.0 | 76.5 ± 0.4 | 77.0 ± 0.3 | 0.8x | 1.2x | 1.6x | 2.0x | 2.4x |
| CAL-DER | 71.5 ± 2.7 | 74.9 ± 3.2 | 75.7 ± 1.5 | 76.9 ± 1.6 | 78.3 ± 0.8 | 1.7x | 2.7x | 3.7x | 4.8x | 5.8x |
| CAL-SD | 73.6 ± 1.9 | 74.9 ± 1.1 | **77.7** ± 1.3 | 76.4 ± 0.3 | 78.0 ± 0.9 | 1.7x | 2.7x | 3.7x | 4.8x | 5.8x |
| CAL-SDS2 | 74.7 ± 2.8 | 75.5 ± 1.0 | 77.2 ± 0.9 | **78.1** ± 0.8 | **79.2** ± 0.5 | 1.4x | 2.1x | 2.9x | 3.7x | 4.5x |
| AL w/ WS | 74.6 ± 0.7 | **76.1** ± 0.4 | 76.3 ± 1.0 | 76.3 ± 1.5 | 77.2 ± 0.9 | 1.2x | 1.7x | 1.6x | 1.8x | 1.8x |
| AL | 73.9 ± 2.9 | 75.5 ± 0.5 | 76.6 ± 2.0 | 76.3 ± 0.9 | 77.3 ± 1.6 | 1.0x | 1.0x | 1.0x | 1.0x | 1.0x |

Table 5: COLA Results.

**Single-Cell Cell-Type Identity** Please refer to table 6 and figure 12.

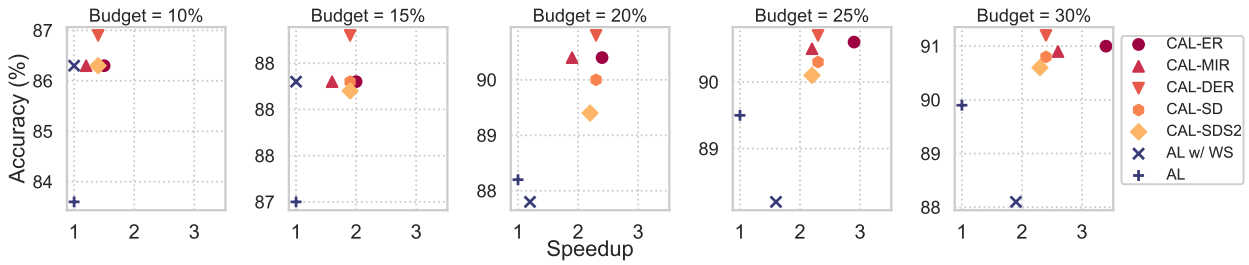


Figure 12: Single-Cell Cell-Type Identity Classification Results

| | Test Accuracy (%) | | | | | Factor Speedup | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| **Method** | **10%** | **15%** | **20%** | **25%** | **30%** | **10%** | **15%** | **20%** | **25%** | **30%** |
| CAL-ER | 86.3 ± 0.1 | 88.3 ± 0.1 | 89.7 ± 0.3 | 90.6 ± 0.2 | 91.0 ± 0.1 | 1.5× | 2.0× | 2.4× | 2.9× | 3.4× |
| CAL-MIR | 86.3 ± 0.1 | 88.3 ± 0.1 | 89.7 ± 0.2 | 90.5 ± 0.2 | 90.9 ± 0.2 | 1.2× | 1.6× | 1.9× | 2.2× | 2.6× |
| CAL-DER | **86.9** ± 0.3 | **88.8** ± 0.3 | **89.9** ± 0.3 | **90.7** ± 0.2 | **91.2** ± 0.1 | 1.4× | 1.9× | 2.3× | 2.8× | 3.3× |
| CAL-SD | 86.3 ± 0.1 | 88.3 ± 0.1 | 89.5 ± 0.2 | 90.3 ± 0.2 | 90.8 ± 0.2 | 1.4× | 1.9× | 2.3× | 2.8× | 3.3× |
| CAL-SDS2 | 86.3 ± 0.1 | 88.2± 0.1 | 89.2 ± 0.3 | 90.1± 0.2 | 90.6± 0.1 | 1.4× | 1.9× | 2.3× | 2.8× | 3.3× |
| AL w/ WS | 86.3 ± 0.1 | 88.3 ± 0.1 | 88.4 ± 0.8 | 88.2 ± 0.8 | 88.1 ± 0.8 | 1.0× | 1.0× | 1.2× | 1.6× | 1.9× |
| AL | 83.6 ± 1.0 | 87.0 ± 0.3 | 88.6 ± 0.1 | 89.5 ± 0.2 | 89.9 ± 0.3 | 1.0× | 1.0× | 1.0× | 1.0× | 1.0× |

Table 6: Single-Cell Cell-Type Identity Classification Results

### A.3 Out-of-distribution (OOD) generalization

Here we show results from the figure 5 in a tabular form in table 7. We compare the performance of CAL methods with baseline AL for OOD generalization. Reported mean values and standard deviations are computed over three different random seeds. As shown in the figure 5 in the main paper, and the table below, CAL methods are as robust to perturbations as baseline AL, despite the speedup. Finally, in table 9 we report that CAL-SDS2 is better compared to other CAL methods, on average.

| **Corruption** | CAL-ER | CAL-MIR | CAL-DER | CAL-SD | CAL-SDS2 | AL | AL w/ WS |
|---|---|---|---|---|---|---|---|
| saturate | 90.4 ± 0.4 | 90.7 ± 0.2 | 90.4 ± 0.1 | 90.5 ± 0.4 | 90.8 ± 0.2 | 90.7 ± 0.4 | 90.8 ± 0.2 |
| impulse noise | 52.4 ± 1.4 | 53.4 ± 0.8 | 53.3 ± 3.9 | 53.6 ± 2.6 | 55.8 ± 1.3 | 54.5 ± 2.9 | 48.7 ± 2.7 |
| defocus blur | 79.6 ± 1.3 | 81.6 ± 0.4 | 79.3 ± 1.2 | 80.6 ± 1.1 | 79.6 ± 1.7 | 80.9 ± 1.0 | 80.4 ± 0.7 |
| contrast | 74.6 ± 0.3 | 76.1 ± 0.8 | 73.8 ± 0.9 | 75.1 ± 1.0 | 74.2 ± 0.6 | 77.1 ± 1.4 | 76.4 ± 2.5 |
| frost | 76.0 ± 0.4 | 76.1 ± 2.0 | 73.8 ± 1.0 | 75.4 ± 1.5 | 76.9 ± 1.2 | 75.4 ± 1.6 | 77.0 ± 0.3 |
| speckle noise | 61.7 ± 1.7 | 61.7 ± 2.6 | 60.9 ± 3.7 | 61.6 ± 3.8 | 64.3 ± 0.5 | 61.7 ± 0.3 | 59.4 ± 1.0 |
| pixelate | 74.4 ± 1.3 | 73.9 ± 2.2 | 75.2 ± 0.9 | 74.8 ± 1.6 | 76.5 ± 0.7 | 75.9 ± 1.3 | 77.0 ± 1.3 |
| zoom blur | 74.4 ± 1.0 | 76.7 ± 1.0 | 74.4 ± 2.3 | 75.4 ± 1.5 | 74.1 ± 2.8 | 75.9 ± 1.9 | 75.3 ± 1.0 |
| elastic transform | 82.5 ± 0.1 | 83.5 ± 0.2 | 82.0 ± 0.5 | 82.1 ± 0.8 | 82.6 ± 0.6 | 82.0 ± 1.3 | 82.9 ± 0.2 |
| spatter | 81.8 ± 0.7 | 82.9 ± 1.2 | 82.4 ± 0.6 | 82.6 ± 0.5 | 82.8 ± 0.8 | 83.0 ± 1.0 | 82.5 ± 1.2 |
| snow | 80.3 ± 0.4 | 80.2 ± 0.7 | 79.5 ± 0.8 | 79.7 ± 0.7 | 80.8 ± 0.1 | 80.1 ± 0.8 | 80.6 ± 0.3 |
| fog | 86.8 ± 0.3 | 86.9 ± 0.3 | 85.9 ± 0.9 | 86.3 ± 0.5 | 86.5 ± 0.4 | 86.6 ± 0.3 | 87.9 ± 0.7 |
| Gaussian noise | 46.6 ± 1.0 | 46.6 ± 3.3 | 44.9 ± 6.3 | 46.7 ± 4.7 | 50.7 ± 0.9 | 45.9 ± 0.4 | 43.2 ± 1.1 |
| brightness | 92.1 ± 0.3 | 92.4 ± 0.3 | 92.1 ± 0.3 | 92.2 ± 0.1 | 92.5 ± 0.1 | 92.7 ± 0.1 | 92.6 ± 0.2 |
| Gaussian blur | 69.7 ± 2.2 | 72.3 ± 1.3 | 69.3 ± 2.5 | 71.3 ± 1.5 | 69.3 ± 2.8 | 71.6 ± 1.6 | 70.4 ± 1.3 |
| motion blur | 75.1 ± 1.2 | 77.1 ± 1.2 | 74.4 ± 0.9 | 75.1 ± 1.6 | 74.5 ± 0.7 | 74.7 ± 0.6 | 76.8 ± 0.4 |
| shot noise | 58.6 ± 1.5 | 58.5 ± 2.9 | 57.3 ± 4.8 | 58.6 ± 4.1 | 61.6 ± 0.5 | 58.3 ± 0.2 | 56.1 ± 1.0 |
| jpeg compression | 79.0 ± 1.5 | 78.4 ± 0.5 | 78.5 ± 0.2 | 78.6 ± 0.5 | 79.1 ± 0.1 | 77.7 ± 0.1 | 77.7 ± 0.1 |
| glass blur | 51.8 ± 1.4 | 54.6 ± 3.0 | 48.6 ± 2.8 | 50.7 ± 2.1 | 53.9 ± 2.4 | 49.4 ± 3.1 | 52.6 ± 2.1 |

Table 7: Accuracy (in %) comparison of CAL methods with the baseline on the CIFAR-10C dataset. Results were reported as an average over three random seeds. Models trained with CAL procedure perform statistically similar to the one trained with baseline AL.

| **Corruption** | CAL-ER/AL | CAL-MIR/AL | CAL-DER/AL | CAL-SD/AL | CAL-SDS2/AL |
|---|---|---|---|---|---|
| saturate | 0.3957 | 0.9104 | 0.2118 | 0.4516 | 0.7961 |
| impulse noise | 0.3209 | 0.5692 | 0.7029 | 0.7134 | 0.5374 |
| defocus blur | 0.2477 | 0.2794 | 0.1561 | 0.7854 | 0.3178 |
| contrast | **0.0335** (+) | 0.3023 | **0.0256** (-) | 0.1020 | **0.0250** (-) |
| frost | 0.5793 | 0.6639 | 0.2147 | 0.9548 | 0.2778 |
| speckle noise | 0.9873 | 0.9922 | 0.7409 | 0.9739 | **0.0017** (+) |
| pixelate | 0.2243 | 0.2521 | 0.4661 | 0.3913 | 0.5636 |
| zoom blur | 0.2824 | 0.5567 | 0.4279 | 0.7238 | 0.3910 |
| elastic transform | 0.5482 | 0.1295 | 0.9975 | 0.9154 | 0.5176 |
| spatter | 0.1504 | 0.9401 | 0.4310 | 0.5843 | 0.7800 |
| snow | 0.7823 | 0.8905 | 0.3747 | 0.5455 | 0.1976 |
| fog | 0.4715 | 0.2534 | 0.2781 | 0.4512 | 0.7142 |
| gaussian noise | 0.3514 | 0.7485 | 0.7836 | 0.8056 | **0.0013** (+) |
| brightness | **0.0309** (-) | 0.2042 | **0.0341** (-) | **0.0091** (-) | **0.0474** (-) |
| gaussian blur | 0.2969 | 0.5867 | 0.2605 | 0.8658 | 0.2997 |
| motion blur | 0.6474 | **0.0368** (+) | 0.6263 | 0.6801 | 0.7135 |
| shot noise | 0.7252 | 0.8920 | 0.7311 | 0.9091 | **0.0004** (+) |
| jpeg compression | **0.0110** (+) | 0.1080 | **0.0016** (+) | **0.0286** (+) | **0.0000** (+) |
| glass blur | 0.2820 | 0.1047 | 0.7633 | 0.5807 | 0.1182 |

Table 8: Pairwise p-values were computed to compare each Confidence-Aware Learning (CAL) method with the standard Active Learning (AL) approach across various corruption tests. p-values less than 0.05 are highlighted in **bold**, and instances, where CAL methods outperform (+) or underperform (-) compared to standard AL, are indicated. Overall, our findings suggest that the disparities between CAL and AL methods are typically not statistically significant, implying that CAL generally does not compromise robustness. When a statistically significant difference does arise, CAL (particularly CAL-SDS2) tends to outperform AL more frequently than it falls short.

### A.4 Correlation between CAL and baseline AL

In figure 13, we provide the Pearson correlation between the uncertainty scores of models on the held-out test set, before every query round. A positive correlation between CAL models and baseline AL models

| Method | Average Accuracy difference (in %) |
|---|---|
| CAL-ER | -0.34 |
| CAL-MIR | 0.49 |
| CAL-DER | -0.96 |
| CAL-SD | -0.17 |
| CAL-SDS2 | 0.63 |
| AL w/ WS | -0.32 |

Table 9: The average difference in the accuracy (in %) of models trained with baseline AL from models trained with CAL, on different benchmarks across the CIFAR10-C dataset. Higher numbers are better. We can see that CAL-SDS2 is better compared to other CAL methods, on average. We hypothesize that this can be attributed to submodular sampling from history.

before every query round suggests that the nature of examples chosen by the CAL models is similar to that of baseline AL models. Note that each entry in the correlation matrix is averaged over three random seeds corresponding to the random initialization of each model.

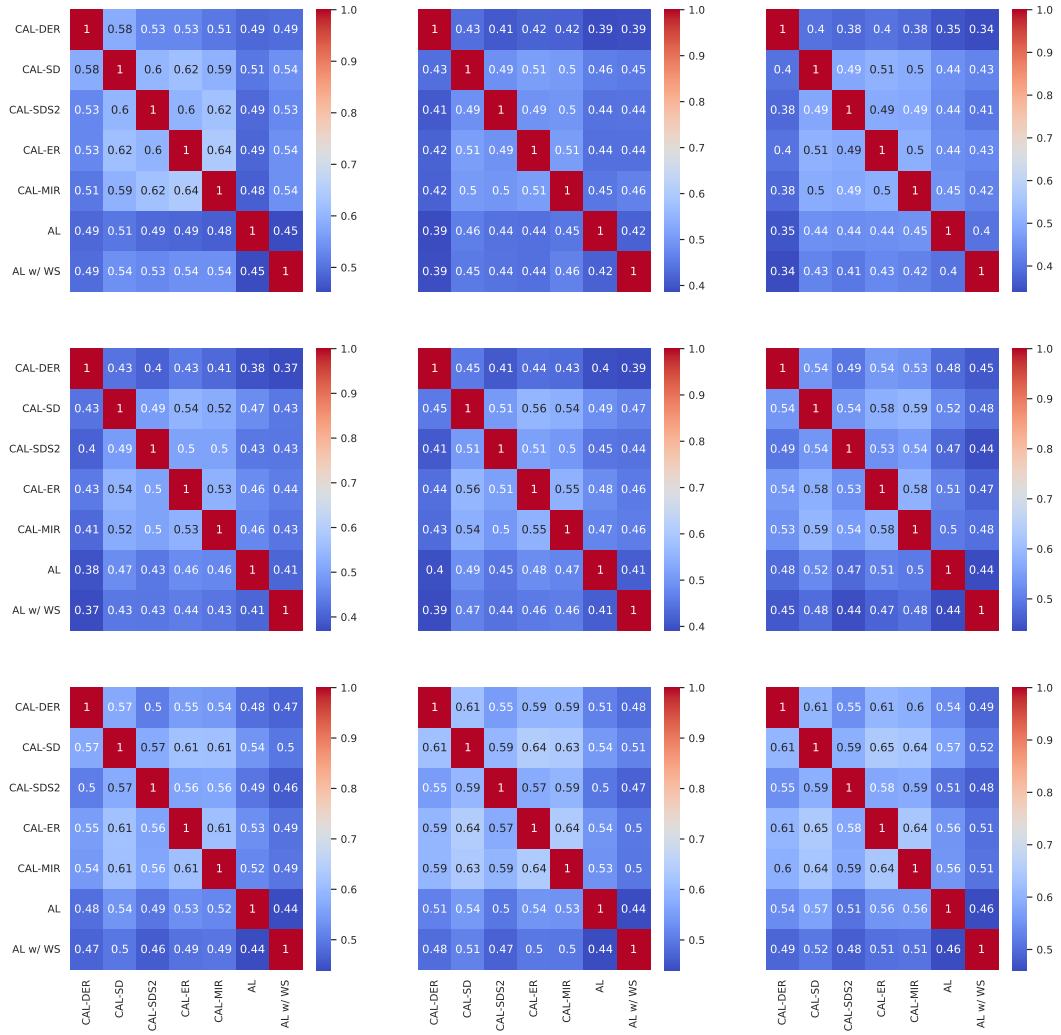


Figure 13: Pearson correlation between uncertainty scores of models on held out set after every query round. We are showing nine query rounds in row-major order (so the top left is after the first query round, the top middle is after the second, and so on). Positive correlation of uncertainty scores suggests that the nature of examples the models are uncertain about, and thus likely to be chosen at every query round, is similar between the CAL-trained models and baseline AL-trained models.

### A.5 Hyperparameters

For every dataset and every CAL/AL strategy, learning rate ($lr$) and batch size ($m$) are chosen based on whichever setting achieves the highest performance on standard AL. This means CAL methods can still improve if we tune either of learning rate or batch size. Replay size is critical for the performance of continual learning algorithms. On the one hand, we do not want the replay batch size, $m^{(h)}$, to be too small since then we will forget some history. But on the other hand, we also do not want $m^{(h)}$ to be too large since there will be a computational cost associated with that. Considering the mentioned constraints, for all CAL methods, we, therefore, set replay size as $m^{(h)} \in \{m, 2m\}$ (used in all CAL methods). Via experimentation on a subset of the datasets, we found $m^{(h)}$ less than or greater than this range suffered either from forgetting (when too small) or extra computation without an accuracy benefit (when too large). We set $\alpha \in \{0.1, 0.25, 0.5, 0.75\}$ (used in CAL-DER, CAL-SD, and CAL-SDS2), $\beta \in \{0.75, 1\}$ (used in CAL-DER). The scale and size of the search space for $\alpha$ and $\beta$ is inspired from Buzzega et al. (2020). Lastly, $\sigma \in \{0.1, 1\}$ (used in CAL-SDS2), and $\lambda \in \{0.1, 1, 10\}$ (used in CAL-SDS2).

We select the configuration for each CAL model that achieves the highest accuracy. "$c$" is the hyperparameter used in CAL-MIR and CAL-SDS2 to subsample the history before finding the $m^{(h)}$ samples to replay, but this parameter is not tuned for any of the presented results. We list the specific set of hyperparameters we use for all the main experimental results in this section.

#### A.5.1 FMNIST

All experiments for FMNIST used a ResNet-18 with an SGD optimizer, with learning rate of 0.01 and batch size of 64. For all the CAL methods, we fix $m^{(h)} = 128$. A NVIDIA GeForce RTX 1080 GPU was used to run all the reported experiments.

**CAL-MIR** $c = 256$

**CAL-DER** $\alpha = 0.1$, $\beta = 1$

**CAL-SD** $\alpha = 0.25$

**CAL-SDS2** $c = 256$, $\alpha = 0.25$, $\sigma = 0.1$, $\lambda = 1$

#### A.5.2 CIFAR-10

All experiments for CIFAR-10 used a ResNet-18 with an SGD optimizer, with learning rate of 0.02 and a batch size of 20. For all the CAL methods, we fix $m^{(h)} = 40$. Training is done on an NVIDIA GeForce RTX 2080.

**CAL-MIR** $c = 100$

**CAL-DER** $\alpha = 0.1$, $\beta = 1$

**CAL-SD** $\alpha = 0.25$

**CAL-SDS2** $c = 100$, $\alpha = 0.25$, $\sigma = 0.1$, $\lambda = 0.1$

#### A.5.3 MedMNIST

All experiments for MedMNIST used a ResNet-18 with an Adam optimizer, with learning rate of 0.001 and a batch size of 128. For all CAL methods, we fix $m^{(h)} = 128$. All reported models were trained on an NVIDIA GeForce RTX 2080.

**CAL-MIR** $c = 270$

**CAL-DER** $\alpha = 0.1$, $\beta = 1$

**CAL-SD** $\alpha = 0.5$

**CAL-SDS2** $c = 270$, $\alpha = 0.5$, $\sigma = 0.1$, $\lambda = 10$

#### A.5.4 Amazon Polarity Review

Throughout our experiments, we sample 2M sentences and use them as the total training set instead. We use Adam optimizer with default parameters with a learning rate of 0.001 and a batch size of 128 for 6 epochs. For all the CAL methods, we fix $m^{(h)} = 128$. All reported models were trained on an NVIDIA GeForce 1080 Ti.

**CAL-MIR** $c = 256$,

**CAL-DER** $\alpha = 0.25$, $\beta = 0.75$

**CAL-SD** $\alpha = 0.5$

**CAL-SDS2** $c = 256$, $\alpha = 0.75$, $\sigma = 1$, $\lambda = 1$

#### A.5.5 COLA

For all of our experiments we use Huggingface's transformer library Wolf et al. (2020) and use a maximum sentence length of 100. We use Adam optimizer and a learning rate of $5 \cdot 10^{-5}$, use a batch size of 25 and $m^{(h)} = 25$. Models were trained on a single NVIDIA GeForce 1080 Ti.

**CAL-MIR** $c = 50$

**CAL-DER** $\alpha = 0.25$, $\beta = 0.75$

**CAL-SD** $\alpha = 0.75$

**CAL-SDS2** $c = 50$, $\alpha = 0.5$, $\sigma = 1$, $\lambda = 1$.

#### A.5.6 Single-Cell Cell-Type Identity Classification

All experiments use SGD optimizer with standard parameters with learning rate of 0.001 and a batch size 128. For all the CAL methods, we fix $m^{(h)} = 128$. Training is done on an NVIDIA A100-PCIE-40GB.

**CAL-MIR** $c = 200$,

**CAL-DER** $\alpha = 0.1$, $\beta = 1$

**CAL-SD** $\alpha = 1$

**CAL-SDS2** $c = 100$, $\alpha = 0.25$, $\sigma = 0.1$, $\lambda = 1$

### A.6 Hyperparameter Sensitivity Analysis for Proposed Methods

In this section, we provide a hyperparameter sensitivity analysis methods for CAL-SD and CAL-SDS2 on CIFAR-10. For CAL-SD, only $\alpha$ is tuned and the best value is used for CAL-SDS2. Therefore, we tuned 1 hyperparameter for CAL-SD while $\sigma$ and $\lambda$ are tuned for CAL-SDS2. Note that the differences in final test accuracy at 50% budget on CIFAR-10 across different configurations are negligible, as shown in the tables below.

| $\alpha$ | 0.1 | 0.25 | 0.75 | 0.9 |
|---|---|---|---|---|
| | 93.79 | 93.90 | 93.58 | 93.38 |

Table 10: CAL-SD Sensitivity

| $\sigma$ \ $\lambda$ | 0.1 | 0.5 | 1 | 5 | 10 |
|---|---|---|---|---|---|
| 0.1 | 94.20 | 94.27 | 94.32 | 94.34 | 94.16 |
| 1 | 94.33 | 94.22 | 94.28 | 94.15 | **94.44** |
| 10 | 94.32 | 94.29 | 94.27 | 94.30 | 94.18 |

Table 11: CAL-SDS2 Sensitivity

## B A primer on Continual Learning

We define $\mathcal{D}_{1:n} = \bigcup_{i\in[n]} \mathcal{D}_i$. In CL, the dataset consists of $T$ tasks $\{\mathcal{D}_1, ..., \mathcal{D}_T\}$ that are presented to the model sequentially, where $\mathcal{D}_t = \{(x_i, y_i)\}_{i\in N_t}$, $N_t$ are the task-$t$ sample indices, and $n_t = |N_t|$. At time $t \in [T]$, the data/label pairs are sampled from the current task $(x, y) \sim \mathcal{D}_t$, and the model has only limited access to the history $\mathcal{D}_{1:t-1}$. If the model is trained only on $\mathcal{D}_t$ using standard optimization algorithms, the model will exhibit catastrophic forgetting. The CL objective is to efficiently adapt the model to $\mathcal{D}_t$ while retaining the performance on the history. Given a loss function $\ell : \mathcal{X} \times \mathcal{Y} \mapsto \mathbb{R}$, initial parameters $\theta_{t-1}$, and a model $f$, $\theta_t$ can be obtained as the solution to the CL optimization problem (Aljundi et al., 2019b; Chaudhry et al., 2019; Lopez-Paz & Ranzato, 2017):

$$\min_{\theta} \mathop{\mathbb{E}}_{(x,y)\sim\mathcal{D}_t} \ell(y, f(x;\theta))$$
$$\text{s.t.} \mathop{\mathbb{E}}_{(x',y')\sim\mathcal{D}_{1:t-1}} \ell\Big(y', f(x';\theta))\Big) \leq \mathop{\mathbb{E}}_{(x',y')\sim\mathcal{D}_{1:t-1}} \ell\Big(y', f(x';\theta_{t-1}))\Big)$$

## C Results for Additional Active Learning Strategies

In this section, we demonstrate that CAL methods can accelerate AL strategies other than entropy sampling without incurring any significant performance drops. We test multiple AL strategies on FMNIST Xiao et al. (2017) and CIFAR-10 Krizhevsky (2009). Note that the speedups are approximately the same as the ones reported in Section A since the training time is generally independent of the selected AL strategy.

### C.1 Overview of Strategies

**Margin Score Sampling** This strategy is another form of uncertainty sampling Settles (2009) as described in the main paper. Instead of the entropy of $f(x;\theta)$, the margin score is used as the entropy score i.e., $h(x) \triangleq 1 - (f(x;\theta)_i - f(x;\theta)_j)$ where $i$ and $j$ are the indices corresponding to the highest and second highest values of $f(x;\theta)$ respectively.

**FASS** FASS Wei et al. (2015) is a two-staged selection method that uses both uncertainty sampling and submodular maximization. Initially, a set of samples $\mathcal{A}$ of cardinality $c * b_t$ is chosen from $\mathcal{U}$ using uncertainty sampling, where $c > 1$ is a tuneable hyperparameter. Next, $U_t$ is constructed by greedily selecting samples that maximize a submodular set function $G : 2^{\mathcal{A}} \to \mathbb{R}_+$ defined on a ground set $\mathcal{A}$. Entropy is once again used as the uncertainty metric for the initial stage. For the second stage, $G$ is defined to be the facility location function Wei et al. (2015) expressed below:

$$G(\mathcal{S}) = \sum_{x_i\in\mathcal{A}} \max_{x_j\in\mathcal{S}} w_{ij}, \tag{4}$$

where $\mathcal{S} \subseteq \mathcal{A}$ and $w_{ij}$ is a similarity score between samples $x_i$ and $x_j$. In our experiments, $w_{ij} = \exp\left(-\|z_i - z_j\|^2/2\sigma^2\right)$ where $z_i$ is the penultimate layer representation of model $f$ for $x_i$ and $\sigma$ is a hyperparameter.

**GLISTER** GLISTER Killamsetty et al. (2021a) solves a bi-level optimization problem in order to select samples to label. Specifically, GLISTER solves

$$\operatorname*{argmax}_{\mathcal{S}\subseteq\mathcal{U}_t, |\mathcal{S}|\leq b_t} LL_V(\operatorname*{argmax}_{\theta} LL_T(\theta, \mathcal{S}), \mathcal{V}) \tag{5}$$

where $LL_V$ is the log-likelihood on the validation set $\mathcal{V}$, and $LL_T$ is the log-likelihood on the subset $\mathcal{S}$.

### C.2 Results

| | Test Accuracy (%) | | | | |
|---|---|---|---|---|---|
| **Method** | **10%** | **15%** | **20%** | **25%** | **30%** |
| CAL-ER | **92.8** ± 0.1 | **94.1** ± 0.1 | 94.8 ± 0.1 | **95.1** ± 0.3 | **95.2** ± 0.2 |
| CAL-MIR | 92.6 ± 0.2 | **94.1** ± 0.4 | **94.9** ± 0.2 | 95.0 ± 0.2 | **95.2** ± 0.2 |
| CAL-DER | 91.8 ± 0.5 | 93.1 ± 0.1 | 94.3 ± 0.3 | 94.6 ± 0.1 | 94.8 ± 0.2 |
| CAL-SD | 92.5 ± 0.1 | 93.8 ± 0.1 | 94.8 ± 0.0 | **95.1** ± 0.2 | **95.2** ± 0.0 |
| CAL-SDS2 | 87.8 ± 1.1 | 93.4 ± 0.1 | 94.6 ± 0.1 | 95.0 ± 0.2 | **95.2** ± 0.1 |
| AL w/ WS | **92.8** ± 0.0 | 94.0 ± 0.3 | 94.6 ± 0.1 | 94.8 ± 0.1 | 95.0 ± 0.2 |
| AL | 92.7 ± 0.1 | **94.1** ± 0.3 | **94.9** ± 0.1 | 95.0 ± 0.2 | **95.2** ± 0.1 |

Table 12: FMNIST with Margin Score Sampling

| | Test Accuracy (%) | | | | |
|---|---|---|---|---|---|
| **Method** | **10%** | **20%** | **30%** | **40%** | **50%** |
| CAL-ER | 81.5 ± 0.1 | 89.3 ± 0.1 | 92.2± 0.2 | 93.4± 0.1 | 93.8 ± 0.0 |
| CAL-MIR | 81.9 ± 0.1 | 89.6 ± 0.2 | 92.2 ± 0.4 | 93.6 ± 0.0 | 94.0 ± 0.2 |
| CAL-DER | 83.0 ± 0.2 | 89.5 ± 0.2 | 92.2 ± 0.2 | 93.2 ± 0.2 | 93.6 ± 0.0 |
| CAL-SD | 82.6 ± 0.4 | 89.9 ± 0.4 | 92.4 ± 0.2 | 93.5 ± 0.1 | 93.8 ± 0.2 |
| CAL-SDS2 | 82.5 ± 0.2 | 90.2 ± 0.2 | 92.5 ± 0.2 | **93.8** ± 0.2 | **94.1** ± 0.1 |
| AL w/ WS | **83.1** ± 0.1 | **90.3** ± 0.3 | **93.0** ± 0.2 | 93.5 ± 0.3 | 93.6 ± 0.2 |
| AL | 75.1 ± 1.2 | 87.1 ± 1.0 | 90.2± 0.5 | 92.0± 0.0 | 92.8 ± 0.5 |

Table 13: CIFAR-10 with Margin Score Sampling

| | Test Accuracy (%) | | | | |
|---|---|---|---|---|---|
| **Method** | **10%** | **15%** | **20%** | **25%** | **30%** |
| CAL-ER | 92.6 ± 0.1 | **93.9** ± 0.2 | 94.6 ± 0.2 | **95.0** ± 0.1 | 94.9 ± 0.0 |
| CAL-MIR | 92.5 ± 0.1 | 93.8 ± 0.3 | 94.6 ± 0.1 | 94.8 ± 0.1 | 94.9 ± 0.2 |
| CAL-DER | 92.7 ± 0.1 | 93.8 ± 0.1 | 94.5 ± 0.1 | 94.7 ± 0.1 | **95.0** ± 0.2 |
| CAL-SD | **92.8** ± 0.1 | **93.9** ± 0.1 | **94.7** ± 0.1 | 94.8 ± 0.3 | 94.9 ± 0.1 |
| CAL-SDS2 | **92.8** ± 0.0 | 93.8 ± 0.2 | 94.5 ± 0.1 | 94.8 ± 0.2 | 94.9 ± 0.1 |
| AL w/ WS | 92.5 ± 0.1 | 93.8 ± 0.3 | 94.0 ± 0.2 | 94.3 ± 0.2 | 94.3 ± 0.0 |
| AL | 92.7 ± 0.4 | **93.9** ± 0.1 | 94.5 ± 0.1 | 94.7 ± 0.3 | 94.8 ± 0.1 |

Table 14: FMNIST with FASS

| | Test Accuracy (%) | | | | |
|---|---|---|---|---|---|
| **Method** | **10%** | **20%** | **30%** | **40%** | **50%** |
| CAL-ER | 82.2 ± 0.2 | 89.8 ± 0.2 | 92.5± 0.2 | 93.4± 0.4 | 93.7 ± 0.2 |
| CAL-MIR | 82.2 ± 0.3 | 89.4 ± 0.2 | 92.3 ± 0.1 | 93.4 ± 0.0 | 93.5 ± 0.1 |
| CAL-DER | **83.1** ± 0.3 | 89.7 ± 0.2 | 91.9 ± 0.1 | 93.1 ± 0.2 | 93.5 ± 0.1 |
| CAL-SD | 83.0 ± 0.3 | 90.0 ± 0.3 | 92.5 ± 0.1 | 93.5 ± 0.1 | **94.0** ± 0.1 |
| CAL-SDS2 | 83.0 ± 0.1 | 90.1 ± 0.1 | 92.7 ± 0.2 | 93.5 ± 0.2 | **94.0** ± 0.0 |
| AL w/ WS | 82.8 ± 0.4 | **90.3** ± 0.1 | **92.8** ± 0.2 | **93.6** ± 0.1 | 93.7 ± 0.3 |
| AL | 72.5 ± 2.0 | 86.6 ± 0.4 | 90.1± 0.4 | 91.7± 0.2 | 92.9 ± 0.2 |

Table 15: CIFAR-10 with FASS

| | Test Accuracy (%) | | | | |
|---|---|---|---|---|---|
| **Method** | **10%** | **15%** | **20%** | **25%** | **30%** |
| CAL-ER | 92.6 ± 0.0 | **93.9** ± 0.2 | 94.3 ± 0.1 | **94.7** ± 0.1 | 94.7 ± 0.2 |
| CAL-MIR | 92.5 ± 0.0 | **93.9** ± 0.4 | 94.3 ± 0.2 | 94.4 ± 0.2 | 94.6 ± 0.1 |
| CAL-DER | **92.7** ± 0.1 | **93.9** ± 0.2 | 94.3 ± 0.3 | **94.7** ± 0.2 | **94.9** ± 0.3 |
| CAL-SD | 92.6 ± 0.1 | 93.8 ± 0.1 | **94.4** ± 0.3 | 94.6 ± 0.1 | 94.7 ± 0.1 |
| CAL-SDS2 | 92.6 ± 0.1 | **93.9** ± 0.2 | **94.4** ± 0.2 | 94.6 ± 0.3 | 94.7 ± 0.2 |
| AL w/ WS | 92.5 ± 0.1 | 93.6 ± 0.1 | 93.9 ± 0.1 | 94.1 ± 0.1 | 94.3 ± 0.1 |
| AL | 92.5 ± 0.2 | 93.8 ± 0.1 | 94.2 ± 0.1 | 94.6 ± 0.2 | 94.7 ± 0.2 |

Table 16: FMNIST with GLISTER

| | Test Accuracy (%) | | | | |
|---|---|---|---|---|---|
| **Method** | **10%** | **20%** | **30%** | **40%** | **50%** |
| CAL-ER | 81.7 ± 0.3 | 89.2 ± 0.2 | 91.9 ± 0.2 | 93.0 ± 0.1 | 93.3 ± 0.1 |
| CAL-MIR | 81.6 ± 0.3 | 89.3 ± 0.4 | 91.7 ± 0.2 | 92.9 ± 0.1 | 93.5 ± 0.2 |
| CAL-DER | **82.8** ± 0.4 | 89.5 ± 0.4 | 91.7 ± 0.4 | 92.8± 0.6 | 93.1 ± 0.2 |
| CAL-SD | 82.5 ± 0.3 | **89.6** ± 0.2 | **92.1** ± 0.2 | 93.1 ± 0.2 | 93.8 ± 0.1 |
| CAL-SDS2 | 81.4 ± 0.4 | 89.1 ± 0.2 | **92.1** ± 0.2 | **93.2** ± 0.3 | **93.9** ± 0.1 |
| AL w/ WS | 81.7 ± 0.4 | 89.3 ± 0.4 | **92.1** ± 0.3 | 93.0 ± 0.1 | 93.3 ± 0.4 |
| AL | 81.0 ± 0.6 | 88.5 ± 0.5 | 91.5 ± 0.3 | 93.0 ± 0.2 | 93.4 ± 0.3 |

Table 17: CIFAR-10 with GLISTER

### C.3 Effect of Query Size

In this section, we demonstrate that speedups with CAL methods can be realized with different query sizes. We demonstrate this by comparing AL with CAL-SDS2 on CIFAR-10, with three different choices of query sizes. We use entropy based uncertainty sampling as the acquisition function. We observe that CAL-SDS2 achieves comparable performance to standard AL with different query sizes, and the speedup can increase when the query size is reduced.

| | | Test Accuracy (%) | | | | | Factor Speedup | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **Query Size** | **Method** | **10%** | **20%** | **30%** | **40%** | **50%** | **10%** | **20%** | **30%** | **40%** | **50%** |
| 5000 | AL | 79.1 | 88.6 | 91.7 | 93.5 | 93.8 | 1.0x | 1.0x | 1.0x | 1.0x | 1.0x |
| | CAL-SDS2 | 79.1 | 89.4 | 92.4 | **94.0** | 94.3 | 1.0x | 1.1x | 1.4x | 1.7x | 2.0x |
| 2500 | AL | 81.9 | 89.6 | 92.4 | 93.5 | 94.1 | 1.0x | 1.0x | 1.0x | 1.0x | 1.0x |
| | CAL-SDS2 | 82.5 | 90.1 | 92.9 | 94.0 | **94.4** | 1.1x | 1.6x | 2.1x | 2.7x | 3.4x |
| 1000 | AL | 84.4 | **90.4** | **93.2** | 93.8 | 94.2 | 1.0x | 1.0x | 1.0x | 1.0x | 1.0x |
| | CAL-SDS2 | **84.3** | 89.8 | 92.5 | 93.2 | 93.5 | 1.6x | 3.1x | 4.2x | 5.3x | 6.4x |

Table 18: Effect of Query Size on CIFAR-10

## D Additional Details on Single-Cell Cell-Type Identity Classification Dataset

The human cell landscape (HCL) dataset consists of scRNA-seq data for 562,977 cells across 63 cell types represented in 56 human tissues. Each cell type may be present in multiple tissues. The cell type classes are highly imbalanced, with the rarest cell type, human embryonic stem cell, accounting for 0.06 % of the total dataset and the most common, fibroblast, accounting for 6%. The raw data is first normalized for library size and scaled to 10000 reads in total, followed by log transformation. We visualize the dataset using UMAP 14.

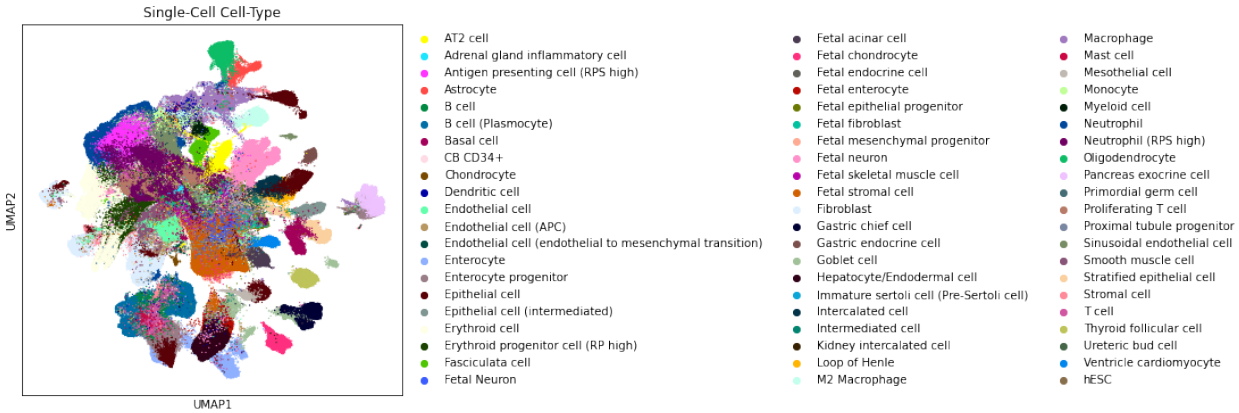


Figure 14: UMAP embedding of single cells in HCL annotated by their cell type.